Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Kevät 2016

INFORMAATIOTIETEIDEN YKSIKKÖ
TAMPEREEN YLIOPISTO

INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 44/2016

TAMPERE 2016

TAMPEREEN YLIOPISTO
INFORMAATIOTIETEIDEN YKSIKKÖ
INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 44/2016
KESÄKUU 2016

Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Kevät 2016

INFORMAATIOTIETEIDEN YKSIKKÖ
33014 TAMPEREEN YLIOPISTO

ISBN 978-952-03-0160-6 (pdf)

ISSN-L 1799-8158
ISSN 1799-8158

Sisällysluettelo

Bitcoinista

Osku Heinonen

Tiivistelmä.
Bitcoin on maksamisen markkinoilla suhteellisen uusi tekijä, ja tässä tutkielmassa pyritään perehtymään bitcoinin aluksi monimutkaiselta tuntuviin perustoihin. Tässä tutkielmassa on tarkoitus antaa lukijalle peruskäsitys louhinnasta. Lisäksi pureudutaan bitcoinin ongelmiin, joita on noussut esiin sitä käytettäessä ja tutkittaessa. Tässä tutkielmassa käsitellään myös bitcoinin etuja ja sen tulevaisuudenkuvaa.

Bitcoin on hyvin nuori keksintö, mutta silti se vaikuttaa melko luotettavalta. Sen ongelmat ovat samankaltaisia kuin käteisen rahan, joka on ollut käytössä jo tuhansia vuosia ja on todettu toimivaksi. Kuitenkin bitcoinilla on havaittu olevan myös ongelmia. Etenkin bitcoinin perusteisiin, eli louhintaan ja käyttäjien anonymiteettiin, liittyvät ongelmat tulisi ratkaista. Bitcoinilla on mahdollisuus nousta tulevaisuudessa suureksi tekijäksi, mutta se vaatii aikaa, jotta mahdolliset käyttäjät oppivat bitcoinin perustat. Lisäksi bitcoinin täytyy kehittyä jatkuvasti, ettei se olisi vaarassa.

Avainsanat: bitcoin, kryptovaluutta, virtuaalivaluutta, louhinta, anonymiteetti

1. Johdanto

Kryptovaluutta on moderni, digitaalinen vaihdon väline, jossa käytetään hyväksi kryptografiaa hallitsemaan rahan luomista ja siirtämistä [Baur *et al*. 2015]. Kryptovaluutta varastoidaan tietokoneelle, kännykälle tai tabletille virtuaaliseen lompakkoon. Tällainen valuutta toimii ainoastaan virtuaalisesti, eikä sitä hallitse mikään pankki tai yksittäinen taho, joka määräisi kryptovaluutan arvon. Arvo sen sijaan määräytyy täysin kysynnän ja tarjonnan mukaan. Tunnetuin kryptovaluutta on bitcoin, ja vuonna 2015 sen arvo vaihteli 203 ja 461 dollarin välillä. Bitcoinia ei kuitenkaan voida pitää virallisena valuuttana, koska se ei ole minkään keskuspankin liikkeelle laskema eikä tukema. Tässä tutkielmassa käytetään kryptovaluutasta synonyymina sanaa virtuaalivaluutta.

Satoshi Nakamoto [2008] esitti idean täysin vertaisverkostoidusta elektronisesta rahasysteemistä, bitcoinista, jossa verkossa tehtävät rahansiirrot menisivät suoraa käyttäjältä käyttäjälle kulkematta minkään rahoituslaitoksen kautta. Tämä on bitcoinin suurin etu verrattuna virallisiin valuuttoihin kuten euroon tai dollariin. Pankit veloittavat välityspalkkioita, kun rahaa siirretään tililtä toiselle; näin ei ole bitcoinin tapauksessa. Lisäksi bitcoinien siirto tapahtuu nopeasti verrattuna tavallisen rahan siirtämiseen esimerkiksi eri pankkien välillä.

Tässä tutkielmassa pohditaan bitcoinin hyviä ja huonoja puolia sekä selvitetään niiden avulla, onko bitcoinilla mahdollisuuksia tulevaisuudessa. On tärkeää saada kartoitettua, millainen on bitcoinin luotettavuus ja turvallisuus vaihdon välineenä. Bitcoinilla on huomattu olevan useita ongelmia. Esimerkiksi Boehm ja Pesch [2014] toteavat, että bitcoinia on epäilty käytettävän rahanpesuun. Kuitenkin bitcoinilla on mahdollisuuksia hyvään. Hurlburtin ja Bojanovan [2014] mielestä bitcoinin rahansiirtojen anonymiteetti on hyvä asia, sillä bitcoin toimii kuten käteinen, mikä vähentää identiteettivarkauksia ja luottokorttihuijauksia. Baurin ja muiden [2015] mukaan bitcoininista on tulossa hyvää vauhtia varteenotettava uusi tekijä verkkomaksamisen markkinoilla.

Tämä tutkielma on kirjallisuuskatsaus, jossa käydään läpi bitcoinin historiaa luvussa 2, bitcoinien louhintaa luvussa 3, bitcoiniin liittyviä ongelmia luvussa 4, etuja luvussa 5 sekä bitcoinin nykytilaa ja tulevaisuutta luvussa 6.

2. Bitcoinin historiaa

Bitcoin sai alkunsa siitä, kun salanimellä Satoshi Nakamoto tunnettu ryhmä tai henkilö julkaisi verkossa artikkelin uudesta täysin vertaisverkostoidusta kryptovaluutasta. Tämä artikkeli julkaistiin vuoden 2008 marraskuussa, ja heti seuraavan vuoden tammikuussa bitcoin (BTC) oli toiminnassa. [Bedford 2013]

Parin ensimmäisen vuoden aikana bitcoinin arvon kasvu oli hidasta. Vuoden 2010 kesäkuussa bitcoinin arvo verrattuna USA:n dollariin oli noin viisi senttiä. Mutta runsaat kolme vuotta tästä eteenpäin sen arvo oli kiivennyt jo 105 dollariin. Arvo oli siis noussut 2100-kertaiseksi näiden kolmen vuoden aikana [Bedford 2013]. Korkeimmillaan bitcoinin arvo on ollut 1151 dollaria vuoden 2013 lopulla [Ciaian *et al.* 2016].

Vuonna 2010 eräällä bitcoin-aiheisella keskustelupalstalla nimimerkki laszlo kirjoitti seuraavasti: ”Haluan ilmoittaa, että olen onnistuneesti vaihtanut 10 000 bitcoinia pizzaan.” [Zohar 2015]. Tätä ostosta pidetään ensimmäisenä, joka on tehty käyttäen bitcoineja. Tämän historiallisen pizzan arvoksi tulisi päivämäärän 16.05.2016 kurssilla 10 000 * 403,28 = 4 032 800 euroa. Zohar [2015] kirjoittaakin, että bitcoinin arvon hurja nousu on synnyttänyt monta tarinaa sekä tehnyt monesta bitcoinin alkuvaiheen kannattajasta miljonäärin.

Satoshi Nakamoto piti huolta bitcoinin ohjelmistosta ja piti muihin yhteyttä verkon välityksellä [Bedford 2013]. Bedfordin mukaan tämä tapahtui aina siihen asti, kunnes hän siirsi vastuun koodikannasta muille vuonna 2011 ja katosi. Satoshi Nakamoton identiteetti on edelleenkin mysteeri, eikä hänestä ole kuulunut sitten vuoden 2011.

Lutherin [2015] mukaan vuoden 2015 puolessa välissä bitcoin on hyväksytty laajasti eri liike-elämän aloilla, isoista Internet-jälleenmyyjistä ruoka-

autoihin. Bitcoinit maksuvälineeksi hyväksyvä liiketoiminta auttaa bitcoinia yleistymään esimerkiksi auttamalla ihmisiä ostamaan, myymään, säilyttämään ja vaihtamaan bitcoineja sekä seuraamaan sen hintaa. Luther [2015] lisää myös, että bitcoin on rutiininomaisesti suuren mediahuomion kohteena.

3. Louhinta

Bitcoin-systeemi ylläpitää käyttäjien bitcoin-rahansiirroista julkisesti saatavilla olevaa tilikirjaa, jota kutsutaan *lohkoketjuksi* [Bedford 2013]. Lohkoketju koostuu useista *lohkoista*, joihin on kerätty hyväksytyt bitcoin-tilisiirrot. Lisätäkseen uuden lohkon lohkoketjuun täytyy löytää allekirjoitus, joka linkittää lohkon tilisiirrot edellisiin lohkoihin. Toisin sanoen jokaisella lohkolla on pää- ja häntäarvo, kuten ohjelmointikielten jonoissa. Voidaan ajatella, että lohko on jono tilisiirtoja ja lohkoketju on jono lohkoja [O'Dwyer and Malone 2014].

Lohkojen linkittäminen toisiinsa tapahtuu matemaattisen prosessin, *louhinnan*, avulla. Louhinnan tarkoitus on siis taata käyttäjien rahansiirtojen onnistuminen turvallisesti. Louhinnalla on myös toinen elintärkeä tehtävä. Koska mikään keskitetty auktoriteetti ei laske liikkeelle tai ohjaa bitcoineja, niiden täytyy syntyä louhinnan tuloksena [O'Dwyer and Malone 2014].

Louhijat pyrkivät ratkaisemaan matemaattisen yhtälön, ja heidän onnistuessaan siinä lohko lisätään lohkoketjun jatkoksi ja louhijat palkitaan onnistuneesta lohkon lisäämisestä lohkoketjuun määrätyllä määrällä bitcoineja. Aluksi onnistuneesta yhtälön ratkaisusta saatava *lohkopalkinto* oli 50 bitcoinia, joka puolittuu joka 210 000:s lohko, eli noin neljän vuoden välein. Tällä hetkellä lohkopalkinto on 25 bitcoinia. Bitcoinien määrä ei tule koskaan saavuttamaan 21 miljoonaa puolittumisen seuraamuksena. [Bedford 2013]

3.1. Louhinta-algoritmi

Maaliarvo on 256-bittinen luku, jonka kaikki bitcoin käyttäjät jakavat. Louhinta on onnistunut ja uusi lohko on hyväksytty bitcoin-verkossa, jos SHA-256-tiiviste lohkon päästä on pienempi tai yhtä suuri kuin nykyinen maaliarvo. Mitä pienempi tämä maaliarvo on, sitä vaikeampaa lohkon luominen on. Lohkon luominen ei ole monivaiheinen prosessi, jossa tehdään miljoonia kertoja tiivistettä, vaan se on enemmänkin kuin lottoa. Jokainen tiiviste antaa arvotun numeron nollan ja 256-bittisen numeron maksimiarvon välistä. [Bitcoin Target]

Kun tilisiirto bitcoin-verkossa on hautautunut tarpeeksi monen lohkon alle, ennen sitä toteutetut tilisiirrot voidaan hylätä, jotta saadaan säästettyä tallennustilaa. Jotta tämä saadaan toteutettua ilman, että rikotaan lohkon tiivistettä, tilisiirrot on tiivistetty *merklepuuhun* (engalnniksi Merkle tree tai hash tree) [Nakamoto 2008]. Merklepuu on puurakenne, jossa bitcoinin tapauksessa tili-

siirrot ovat lehtisolmuja. Lehtisolmujen vanhemmat sisältävät tiivisteen tilisiirroista. Lehtisolmujen vanhempien vanhemmat taas sisältävät tiivistetyn arvon jo kertaalleen tiivistetystä tilisiirrosta. Ottamalla tiivistettä alkaen lapsisolmusta päädytään lopulta puun juureen, jota bitcoinin lohkossa kutsutaan *juuren tiivisteeksi*. Ainoastaan juuren tiiviste puusta on sisällytetty lohkon tiivisteeseen, ja näin ollen vanhat lohkot voidaan puristaa tiiviimmäksi katkomalla puun haarat pois [Nakamoto 2008]. Merklepuun toimintaa lohkossa havainnollistaa kohdan 3.3 kuva 1.

Louhinnan onnistumiseen vaikuttaa myös *muuttuja-arvo* (englanniksi nonce value), joka löytämällä saadaan ratkaistua matemaattinen yhtälö [O'Dwyer and Malone 2014]. Muuttuja-arvo on 32-bittinen kenttä, joka sisältyy lohkon päähän. Lohkon päähän sisältyvät myös kentät versiosta, edellisen lohkon pään tiivisteestä, merklepuun juuren tiivisteestä, ajasta ja maaliarvosta [Bitcoin Block hashing algorithm]. Kun otetaan SHA-256 -tiivistetty arvo lohkon päästä, pyritään saamaan arvo, joka on pienempi kuin maali-arvo, ja näin ollen voitetaan määrätty summa bitcoineja [Bitcoin Nonce]. Lohkon päässä olevista kentistä juuri muuttuja-arvoa muutetaan aina, kun uutta tiivistettä kokeillaan. Sen sijaan merklepuun juuren tiiviste muuttuu aina, kun lohkoon tulee uusi tilisiirto [Bitcoin Block hashing algorithm].

3.2. Louhintaan vaikuttava matematiikka

Lohkot saadaan linkitettyä kaavalla $H(B.N) < T$, jossa H on tiiviste, B on lohkon viimeisimmät tilisiirrot, '.' on ketjutusoperaattori, N on muuttuja-arvo ja T on maaliarvo [O'Dwyer and Malone 2014]. Mikä tahansa muutos lohkon dataan, tässä tapauksessa muuttuja-arvon muutos, tekee lohkon tiivisteestä täysin erilaisen. Koska uskotaan, että on mahdotonta ennustaa, mikä kombinaatio bittejä antaa oikean ratkaisun tiivisteestä, monia eri muuttuja-arvoja täytyy kokeilla [Bitcoin Nonce].

Louhinnassa onnistumiseen vaikuttaa myös *vaikeus* (englanniksi difficulty). Nakamoton [2008] mukaan vaikeus määräytyy nollabittien mukaan. Tämä tarkoittaa sitä, että mitä useammalla nollabitillä maaliarvo alkaa, sitä pienempi se on, ja sitä vaikeampaa louhiminen myös on. Esimerkiksi 256-bittisen maaliarvon ollessa 00000000c937983704a7 on se suurempi kuin 000000000003ba27aa20, ja näin ollen ensimmäinen luvuista on vaikeudeltaan helpompi louhia. Tässä esimerkissä 256-bittiset luvut on lyhennetty 20 merkin mittaisiksi, jotta ne olisi helpompi lukea. Jotta louhinta onnistuisi, lohkon pään tiivisteen täytyy olla pienempi tai yhtä suuri kuin maaliarvo [Bedford 2013].

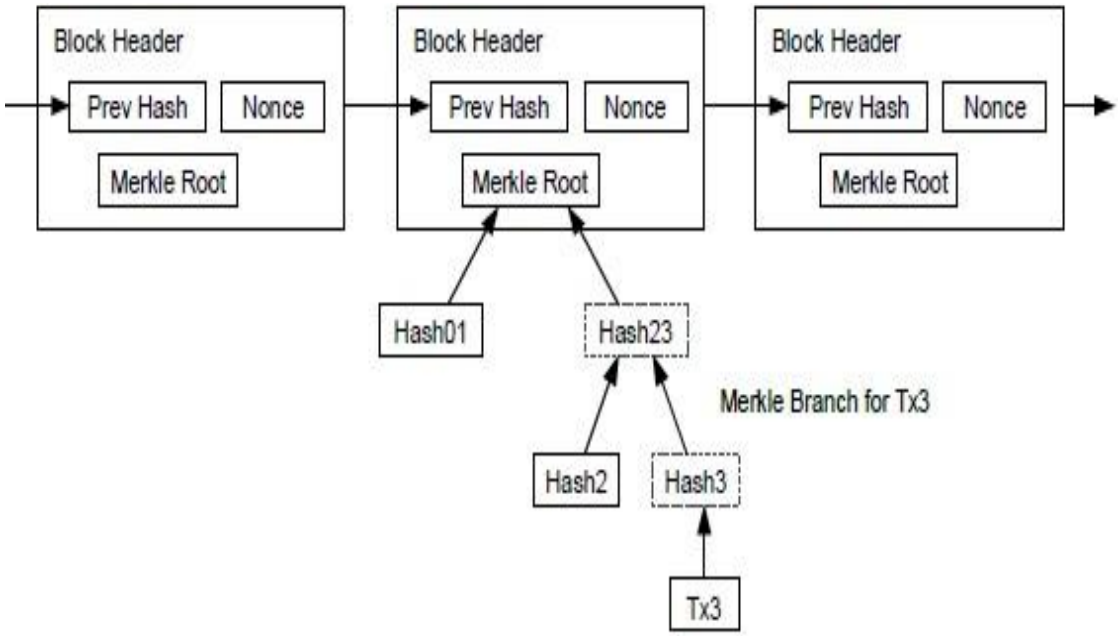
Vaikeuden laskemiseen käytettävä kaava on muotoa $D = T_{max} / T$, jossa D on vaikeus, $T_{max} = (2^{16} - 1)2^{208} \approx 2^{224}$ on maaliarvon suurin mahdollinen arvo. T

on määrätty vaikeus [O'Dwyer and Malone 2014]. Kaavasta voidaan päätellä, että mitä pienempi maaliarvo on, sitä suurempi myös vaikeus on, sillä T_{max} on vakio.

Vaikeus lasketaan uudelleen 2016 lohkon eli noin kahden viikon välein sen takia, että saadaan pidettyä uuden lohkon löytymiseen kuluva aika keskiarvollisesti lähellä kymmentä minuuttia [O'Dwyer and Malone 2014]. Todellisuudessa aika syntyneiden lohkojen välillä on satunnainen, sillä jotkut lohkot luodaan muutamien sekuntien päästä toisistaan, kun taas joidenkin peräkkäisten lohkojen luomiseen kuluu aikaa yli tunti [Bedford 2013].

3.3. Todiste työstä

Kuvassa 1 nähdään, kuinka lohkojen linkittyminen tapahtuu. Kuvassa lohkon pää sisältää merklepuun juuren (Merkle Root), edellisen lohkon pääarvon tiivisteen (Prev Hash) ja muuttuja-arvon (Nonce). Kuvasta 1 nähdään myös, miten merklepuu toimii. Siinä on tiivistetty tilisiirto (Tx3) solmuun Hash3. Siitä eteenpäin on tiivistetty Hash2 ja Hash3 solmuun Hash23. Lopuksi Hash01 ja Hash23 on tiivistetty puun juureksi (Merkle Root).



Kuva 1. Lohkojen linkittyminen ja merklepuu [Nakamoto 2008]

Kuvasta 1 nähdään, että vasemmanpuolimmaisesta lohkon päästä on löydetty sellainen tiiviste, että se alittaa maaliarvon. Tämä tiiviste nimetään edellisen lohkon tiivisteeksi (Prev Hash) keskimmäisessä lohkon päässä. Nyt täytyy löytää sellainen muuttuja-arvo (Nonce), jolla lohkon pään tiiviste alittaa nykyisen maaliarvon. Kun tällainen on löydetty, linkitetään tiivistetty arvo nykyisen lohkon päästä seuraavaan lohkoon taas nimellä edellisen lohkon tiiviste. Koko

onnistunutta prosessia, jossa varmistetaan tilisiirrot ja linkitetään lohkot toisiinsa, kutsutaan *todisteeksi työstä* (englanniksi proof-of-work).

Kun todiste työstä on tehty, lohkoa ei voida muuttaa ilman, että tehdään työ uudestaan. Jos joku pyrkisi muuttamaan haluamaansa lohkoa, hänen täytyisi tehdä työ kaikista lohkoista, jotka on linkitetty lohkoketjuun tämän halutun lohkon jälkeen. [Nakamoto 2008]

Voidaan ajatella, että jokin epäluotettava henkilö tai ryhmä yrittäisi perua jonkin halutun rahansiirron ja näin käyttää uudestaan jo käyttämänsä bitcoinit. Käytännössä tämä on mahdollista, joten Nakamoto [2008] on esittänyt ratkaisun, jottei tällaista rikollista toimintaa tapahtuisi. Ratkaisuna toimivat *luotettavat solmut* (englanniksi honest nodes).

Bitcoin-yhteisö luottaa siihen, että suurin osa tietokoneiden voimasta on rehellisten solmujen hallinnassa. Näin ollen rehellinen ketju kasvaa nopeimmin ja päihittää kilpailevat ketjut. Jotta bitcoin-järjestelmää vastaan hyökkäävä taho voisi aiheuttaa vahinkoa johonkin haluttuun lohkoon, sen täytyisi tehdä halutun lohkon todiste työstä. Tämän jälkeen sen tulisi tehdä kaikkien lohkojen todiste työstä, jotka ovat ilmestyneet halutun lohkon jälkeen. Ja vielä tämän työn jälkeen sen täytyisi ottaa kiinni ja ohittaa rehellisten solmujen tekemä työ. [Nakamoto 2008]

Louhinnan vaiheet ovat seuraavat (solmulla tarkoitetaan tässä bitcoin-verkostoon liitettyä louhintatietokonetta): 1) Uudet tilisiirrot julkistetaan jokaiselle solmulle, 2) Jokainen solmu kerää uudet tilisiirrot lohkoon, 3) Jokainen solmu yrittää suorittaa lohkon todistetta työstä, 4) Kun jokin solmu on ratkaissut lohkon todisteen työstä, solmu lähettää lohkon kaikille solmuille, 5) Solmut hyväksyvät lohkon vain, jos kaikki tilisiirrot ovat valideja, 6) Solmut ilmaisevat hyväksyvänsä lohkon työstämällä lohkoketjun seuraavaa lohkoa käyttämällä hyväksytyn lohkon tiivistettä edellisen lohkon tiivisteenä seuraavassa solmussa. [Nakamoto 2008]

3.4. Louhintakoneisto

SHA-256 -tiiviste on suunniteltu siten, ettei sitä voi kääntää toiseen suuntaan, joten louhintaongelman ratkaisu löydetään usein käyttämällä raakaa voimaa. Raa'alla voimalla tarkoitetaan laitteiston tekemää laskentaa.

Suurimmat rajoittavat tekijät bitcoinin louhinnassa ovat laitteiston *tiivisteen laskentanopeus* (englanniksi hash rate) sekä laitteiston käytön hinta. Laskentanopeus, R, mitataan tyypillisesti miljoonissa tiivisteissä per sekunti tai megatiivisteissä (Mhash/s). Laskentanopeus yhdistettynä laitteiston voiman kulutukseen, P, antaa laitteiston energiatehokkuuden, joka on muotoa E = R / P

(Mhash / J). Tämä antaa hyödyllistä tilastotietoa, jotta voidaan vertailla laitteistojen sopivuutta bitcoinien louhintaan. [O'Dwyer and Malone 2014]

Aluksi louhintaa harrastettiin normaaleilla, tavallisille kuluttajille tarkoitetuilla tietokoneilla. Mutta kun bitcoin kasvatti suosiotaan, tapahtui jonkinlainen kilpavarustelu, kun louhijat pyrkivät parantelemaan tiivisteen laskentanopeuttaan [O'Dwyer and Malone 2014]. Aluksi louhintaa tehtiin tietokoneen prosessoreiden avulla. Prosessoreilla tehty louhinta kuitenkin väistyi ja tilalle tulivat grafiikkaprosessorit [Bedford 2013]. Grafiikkaprosessoreilla pystytään tekemään rinnakkaista laskentaa, joka on bitcoinienkin tapauksessa tehokkaampaa kuin laskeminen tavallisilla prosessoreilla.

Koska grafiikkaprosessorien käyttö alkoi yleistyä, louhijoiden täytyi tutkia muitakin vaihtoehtoja, jotta he pysyisivät muita edellä [O'Dwyer and Malone 2014]. Louhijoiden määrän kasvaessa täytyi myös laitteiston tehokkuuden kasvaa, jotta louhijat saisivat mahdollisen lohkopalkinnon. Seuraavaksi louhijoiden joukossa suosikiksi nousi digitaalinen mikropiiri, jonka pystyi konfiguroimaan valmistuksen jälkeen. Tästä tulee sen englanninkielinen nimitys FPGA eli Field-Programmable Gate Array [Bitcoin FPGA].

FPGAn jäätyä louhijoille tehottomaksi otettiin avuksi sovelluskohtainen mikropiiri (englanniksi ASIC eli Application Specific Integrated Circuit). Bitcoin-louhintaan käytettävät ASIC-mikropiirit on luotu juuri louhintaa varten. Koska ASIC-mikropiireillä toteutettu louhinta päihittää prosessorit, grafiikkaprosessorit sekä FGPA-mikropiirit niin louhinnan nopeudessa kuin tehokkuudessakin, kaikki käytössä olevat bitcoin-louhintalaitteistot hyödyntävät yhtä tai useampaa bitcoin-ASIC-mikropiiriä [Bitcoin ASIC].

Nimi	Tyyppi	Laskentanopeus (Mhash / s)	Energiankulutus (W)	Energiatehokkuus (Mhash / J)
Core i7 920	CPU	19,2	195	0,10
Xeon E5520	CPU	6,5	80	0,08
ATI 4890	GPU	97,1	190	0,511
GT 430	GPU	20,24	49	0,413
Icarus	FPGA	380	19,2	19,79
X6500 FPGA Miner	FPGA	400	17,2	23,25
AntMiner S1	ASIC	180 000	360	500
Avalon Batch 1	ASIC	66 300	620	107

Taulukko 1. Louhintalaitteiston vertailua [Bitcoin Mining hardware comparison; Bitcoin Non-Specialized hardware comparison]

Taulukossa 1 on koottu tietoa eri louhintalaitteistoista ja vertailtu niiden tehoa. Jokaisesta edellä käsitellystä louhintalaitteiston tyypistä (CPU, GPU, FPGA ja ASIC) on otettu taulukkoon 1 kaksi tuotetta. Kaikista näistä on koottu taulukkoon laskentanopeus, energiankulutus ja energiatehokkuus. Kuten taulukosta huomataan, laitteiston kehitys on huomattavaa. Koska ASIC-piirilevyt ovat energiatehokkuudeltaan ylivertaisia, ei muilla laitteistotyypeillä ole enää kannattavaa tehdä louhintaa.

3.5. Louhintaryhmät

Koska bitcoinien louhintaan tarvitaan paljon laitteistoa, yksittäisen ihmisen louhinnan tuottavuus on satunnaista. Tästä johtuen yksittäinen louhija voi liittyä louhintaryhmään (englanniksi pool). Empiirinen aineisto todistaa, että bitcoin-louhijat käyttäytyvät strategisesti ja muodostavat ryhmiä. Kun louhija liittyy ryhmään, se ei muuta louhijan odotettua tuottoa. Ryhmään liittyminen laskee tuoton varianssia ja tekee kuukausittaisista tuloista helpommin ennustettavia [Eyal and Gün Sirer 2014].

Ryhmän onnistuessa luomaan uuden lohkon lohkopalkinto jaetaan jäsenten kesken siten, että jokainen saa sen verran kuin on ryhmään panostanut. Ryhmä pyrkii takamaan, että jäsen saisi lähes saman odotetun summan, minkä yksittäinen louhija saisi itsekseen louhimalla [Zohar 2015]. Eri ryhmien välillä on eri tapoja, miten lohkopalkinnot jaetaan jäsenten kesken.

4. Bitcoinin ongelmat

Tässä luvussa tutustutaan bitcoinin ongelmiin. Eyal ja Gün Sirer [2014] ovat esittäneet louhintaan liittyvän ongelman nimeltä itsekäs louhinta (englanniksi Selfish-Mining), joka käydään läpi kohdassa 4.1. Kohdassa 4.2 pohditaan käyttäjän anonymiteettiin liittyviä ongelmia ja viimeisessä kohdassa 4.3 käsitellään bitcoiniin liittyviä juridisia ongelmia. Bitcoinilla on todettu olevan myös paljon muita ongelmia, mutta tässä tutkielmassa keskitytään näihin kolmeen.

4.1. Louhintaan liittyvät ongelmat; Case Selfish-Mining

Kuten edellä todettiin, bitcoin-protokolla vaatii, että suurin osa louhijoista on rehellisiä. Jos salaliitossa oleva ryhmä louhijoita saa hallintaansa enemmistön louhintavoimasta, valuutta ei ole enää hajautettu ja sen hallinta siirtyy salaliitossa olevalle ryhmälle. On siis tärkeää, että protokolla on suunniteltu niin, ettei louhijoilla ole kannustinta muodostaa suuria salaliittoryhmiä. [Eyal and Gün Sirer 2014]

Eyalin ja Gün Sirerin [2014] mukaan tavanomainen uskomus on pitkään ollut, että bitcoinin louhintaprotokolla on oikeudenmukainen osallistujille ja tur-

vallinen vähemmistöä edustavaa hyökkääjää vastaan. Protokollan uskottiin palkitsevan louhijoita tiukasti sen mukaan, kuinka suuren osan koko louhintavoimasta he omistavat. Louhijan uskottiin saavan saman verran tuottoa isossa louhintaryhmässä kuin hän olisi saanut pienessä ryhmässä. Jos sivuutetaan sovittu kustannus ryhmäoperaatioista ja potentiaalisista skaalaeduista, salaliittolouhijoille ei ole hyödyllistä järjestäytyä koko ajan kasvaviin rehellisiin ryhmiin. Näin ollen rehellisten louhijoiden ryhmäytyminen ei muodosta uhkaa systeemille [Eyal and Gün Sirer 2014].

Tavanomainen uskomus on kuitenkin väärä: bitcoin-protokolla ei ole kannustinkelvollinen. Tämä tarkoittaa sitä, että louhija ei saa parasta tulosta itselleen toimiessaan omien aitojen etujensa mukaan [Eyal and Gün Sirer 2014]. Eyal ja Gün Sirer [2014] esittävät strategian, jota voidaan käyttää siten, että vähemmistöryhmä saavuttaa enemmän tuottoa kuin sen ryhmän kohtuullinen osuus todellisuudessa on. Tämä tarkoittaa sitä, että vähemmistöryhmä saavuttaa tuottoa enemmän kuin ryhmän louhintavoiman suhde on louhintavoiman kokonaisuudesta.

Jokainen louhija voi lisätä pätevän lohkon lohkoketjuun yksinkertaisesti julkaisemalla sen peiteverkossa (englanniksi overlay network) muille louhijoille. Jos kaksi louhijaa luo kaksi lohkoa samaan edeltävään lohkoon, lohkoketju haarautuu kahteen haaraan ja muodostaa puun. Muut louhijat saattavat myöhemmin lisätä uusia päteviä lohkoja kumpaan tahansa haaraan. Haarojen muodostuminen on ei-toivottavaa, sillä louhijoiden täytyy ylläpitää kaikkialla hyväksyttyä ja täysin järjestyksessä olevaa tilisiirtojen joukkoa. Jotta haarautuminen saadaan selvitettyä, protokolla määrää louhijoita valitsemaan pisimmän ketjun ja louhimaan sitä. Jokainen louhija lisää lohkoja pisimpään ketjuun, jonka he tietävät, tai ensimmäiseen, josta he ovat kuulleet, jos on olemassa kaksi yhtä pitkää haaraa [Eyal and Gün Sirer 2014].

Eyalin ja Gün Sirerin [2014] esittelemän strategian, itsekkään louhimisen, idea on se, että ryhmä pitää löytämänsä lohkot yksityisinä, ja yrittää siten tahallisesti tehdä lohkoon haaroja. Rehelliset solmut jatkavat julkisen ketjun louhimista samalla, kun ryhmä louhii omaa yksityistä haaraansa. Jos ryhmä löytää enemmän lohkoja, se kehittää pidemmän johdon julkiseen ketjuun, ja jatkaa näiden uusien lohkojen pitämistä yksityisinä. Kun julkinen haara saavuttaa ryhmän yksityisen haaran pituuden, itsekkäät louhijat paljastavat lohkot omasta yksityisestä ketjustaan julkisuuteen [Eyal and Gün Sirer 2014].

Itsekäs louhintastrategia johtaa rehelliset louhijat, jotka noudattavat bitcoin-protokollaa, hukkaamaan resursseja louhiessaan ilman tarkoitusta. Eyalin ja Gün Sirerin [2014] analyysin mukaan sekä rehelliset, että itsekkäät osapuolet hukkaavat resursseja, mutta rehelliset louhijat hukkaavat suhteessa enemmän.

Itsekkään ryhmän palkinnot ylittävät sen osuuden verkoston louhintavoimasta, mikä antaa itsekkäälle ryhmälle kilpailuvaltin, jolla he voivat houkutella rehellisiä louhijoita liittymään itsekkääseen louhintaryhmään.

Itsekäs louhintaryhmä voi nopeasti kasvaa enemmistöksi. Jos tällainen louhintaryhmä saa enemmistön louhintavoimasta, se voi vaihtaa muunneltuun protokollaan, joka sivuuttaa ryhmän ulkopuolella luodut lohkot. Näin ollen itsekkäästä ryhmästä tulee lohkojen ainoa luoja, ja ryhmä korjaa kaiken louhintatuoton. Eyalin ja Gün Sirerin [2014] mukaan itsekäs louhintaryhmä voi hyväksyä lohkoja muilta tahoilta luodakseen illuusion hajautuksesta, vaikka he samalla säilyttävät mahdollisuuden korjata täyden tuoton tarvittaessa. Kuitenkin valuutan hajautettu luonne on romahtanut, ja itsekkään ryhmän johtaja kontrolloi koko systeemiä.

Bitcoin-louhintaprotokolla ei tule milloinkaan olemaan turvallinen itsekästä louhintaryhmää vastaan, jos ryhmä hallitsee yli kolmasosaa koko verkoston louhintavoimasta. Tällainen ryhmä pystyy aina keräämään louhintatuottoa enemmän kuin sen osuus on, vaikka se häviäisi jokaisen lohkokilpailun verkostossa. Eyalin ja Gün Sirerin [2014] mukaan verkostosta ainakin kaksi kolmasosaa täytyy olla rehellisiä, jotta itsekäs louhinta saadaan pysäytettyä.

4.2. Käyttäjän anonymiteetti

Bitcoinin käyttäjien ja tilisiirtojen anonymiteettia pidetään yhtenä sen suurimmista vahvuuksista, ja näin ollen käyttäjän henkilöllisyyttä ei pitäisi bitcoin-verkossa pystyä paljastamaan. Tämän takia käyttäjän yksityisyyteen liittyvät ongelmat ovat vakavia bitcoin-verkolle.

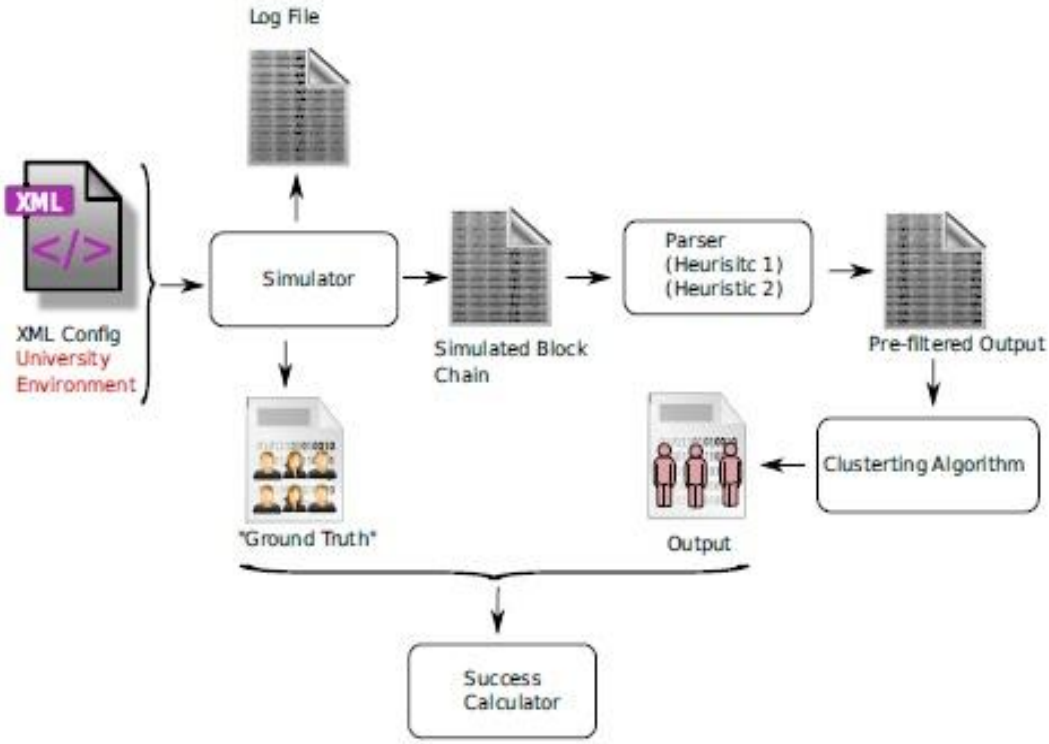
Tilisiirron *viite* on tunniste, joka esittää bitcoin-maksun mahdollisen päämäärän [Bitcoin Address]. Bitcoin-verkossa pystytään helposti ja vapaasti tekemään viitteitä tilisiirroille, joten on mahdollista tehdä uniikki viite jokaiselle tilisiirrolle erikseen. Tämä auttaa käyttäjiä jossain määrin yksityisyytensä suojaamisessa, mutta joitakin tietoja vuotaa silti, vaikka bitcoin-viitteitä ei käytettäisi uudestaan [Zohar 2015]. Androulakin ja muiden [2013] mukaan bitcoinin käyttäjien anonymiteetti ei olekaan niin turvassa kuin on uskottu.

Androulaki ja muut [2013] ovat tehneet tutkimuksen, jossa on arvioitu bitcoinin tarjoamaa yksityisyysturvaa. Tämä arviointi on tehty analysoimalla aitoa bitcoin-systeemiä sekä simuloinnilla, joka luotettavasti matkii bitcoinin käyttöä yliopistossa. Tässä kohdassa keskitytään kuitenkin simulaatioon.

Yliopistoon kehitettyä bitcoin-systeemiä on analysoitu kahdella heuristiikalla, joista toinen keskittyy tapaukseen, jossa tilisiirrolla on useampi syöteviite. Usean syöteviitteen tilisiirto ilmenee, kun käyttäjä yrittää tehdä oston käyttäen viitettä, mutta oston summa ylittää viitteen käytössä olevien bitcoinien arvon.

Tässä tapauksessa bitcoin-asiakasohjelma tekee automaattisesti useamman syöteviitteen samaan tilisiirtoon. Toisin sanoen bitcoin-asiakasohjelma valitsee bitcoinit käyttäjän lompakosta ja suorittaa maksun käyttäen useaa syöteviitettä. Toinen heuristiikoista käsittelee taas *varjoviitteitä*. Varjoviite luodaan automaattisesti bitcoin-asiakasohjelman toimesta ja sen avulla kerätään takaisin vaihtorahat, jotka syntyvät, kun käyttäjä on osallisena tilisiirtoon. [Androulaki *et al*. 2013]

Androulakin ja muiden [2013] mukaan useamman syöteviitteen heuristiikan avulla voidaan päätellä, että jos tilisiirrossa esiintyvillä bitcoineilla on eri viitteet, niin silloin syöteviitteet kuuluvat samalle käyttäjälle. Bitcoin-asiakasohjelmat eivät tarjoa tukea siihen, että eri käyttäjät voisivat osallistua yhteen tilisiirtoon [Adroulaki *et al*. 2013]. Varjoviite heuristiikan perusteella voidaan muodostaa tapaus, jossa bitcoin-tilisiirrolla on kaksi tulosteviitettä, joista toinen on sellainen, jota ei ole esiintynyt bitcoin-verkossa ennen (viite a). Toinen tulosteviite taas vastaa jotakin vanhaa viitettä (viite b). Näistä kahdesta tulosteviitteestä voidaan päätellä, että viite a on viitteen b varjoviite [Androulaki *et al*. 2013].



Kuva 2. Simulaatiokokeen järjestely [Androulaki *et al*. 2013].

Androulaki ja muut [2013] loivat järjestelyn, joka on esitetty kuvassa 2. Kuvasta nähdään, että simulaattori (simulator) ottaa XML-asetukset (englanniksi XML config) syötteenä. XML-asetukset pitävät sisällään käyttäjien määrän, louhijoiden määrän, simuloinnin ajan, lohkon luomisen vaikeuden ja käyttöase-

tukset käyttäjäprofiilien luomiselle sekä bitcoinin myyjille ja ostajille. Simulaattorin tulosteena on loki (englanniksi log file), joka tarkentaa simuloituja tapahtumia. Tulosteena on myös ihmisten kokema totuus (englanniksi ”ground truth”) sekä simuloitu lohkoketju (englanniksi simulated block chain). Perl-pohjainen jäsentäjä käyttää simuloitua lohkoketjua syötteenä ja luokittelee simuloidut viitteet edellä mainittuja heuristiikkoja hyväksikäyttäen. Tästä tuloksena syntyy etukäteen lajiteltu tuloste (englanniksi pre-filtered output), joka syötetään klusterialgoritmiin (englanniksi clustering algorithm), minkä jälkeen sen syötettä vertaillaan koetun totuuden kanssa. Tästä vertailusta saadaan laskettua, kuinka helppo tai vaikea on yhdistää kaksi viitettä toisiinsa tai muodostaa profiileita bitcoin-verkon käyttäjistä.

Simulaatioon osallistuvista 5,2% oli professoreita, 42,0% henkilökuntaa ja 52,8% oppilaita. Simuloinnissa tarkasteltiin kuutta tapahtumaa, joissa jokaisessa oli useita kulutusvaihtoehtoja: syöminen 12 vaihtoehtoa, ruokatavaroiden osto 2 vaihtoehtoa, myyntiautomaatista osto 4 vaihtoehtoa, online ostokset 5 vaihtoehtoa ja kirjojen osto 2 vaihtoehtoa.

Androulakin ja muiden [2013] luomassa simulaatiossa käytettiin kahta eri skenaariota, jossa toisessa vastustaja (joku, jolla on motivaatiota hankkia tietoa viitteistä tai tilisiirroista koskien kaikkia tai osaa bitcoin-käyttäjistä) oli tietoinen kaikkien bitcoin-kauppiaiden palveluiden sijainnista. Toisessa skenaariossa vastustajalla ei ollut tietoa palveluista tai niiden sijainneista, mutta hän oletti, että ainakin 10% tilisiirroista tehtiin Internetin kautta toimitetuista tuotteista. Tapauksessa, jossa 200 käyttäjällä oli 0%:n yksityisyystietoisuus, melkein 42% käyttäjien asetuksista pystyttiin kaappaamaan 80% tarkkuudella. Skenaariossa, jossa käyttäjät olivat täysin tietoisia yksityisyydestänsä, asetusten kaappaaminen väheni 35% samalla tarkkuudella.

Tämän perusteella Androulaki ja muut [2013] päättelevät, että bitcoin-käyttäjien yksityisyys voi olla vaarassa. Käytökseen pohjautuvat klusterointiikat voivat paljastaa jopa 40% käyttäjien profiileista, vaikka käyttäjät manuaalisesti loisivat uusia viitteitä lisätäkseen omaa turvallisuuttaan systeemissä [Androulaki *et al*. 2013].

4.3. Juridiset ongelmat

Tässä kohdassa käsitellään muutamia bitcoinin juridisia ongelmia. Bitcoinilla on enemmänkin juridiikkaan liittyviä ongelmia, mutta tämän työn puitteissa keskitytään bitcoinien varastamiseen alakohdassa 4.3.1 sekä rahanpesuun, veropetoksiin ja laittomaan kaupankäyntiin alakohdassa 4.3.2.

4.3.1 Bitcoinien varastaminen

Jos bitcoinit tai bitcoin-käyttäjät ovat varastamisen kohteena, perinteisen rikosoikeuden menetelmät eivät ole mutkattomia ja laillinen korvaus on epäselvä. Bitcoinit ovat tietokoneella kehitettyjä eivätkä ole fyysisesti olemassa. Tällaiset aineettomat objektit eivät automaattisesti kuulu kansallisten rikosoikeudellisten menetelmien piiriin, jotka suojaavat ryöstöjä vastaan. Esimerkiksi Saksassa vain fyysiset olemassa olevat oliot voivat olla ryöstön kohteena. Muut lailliset menetelmät suojaavat datan manipulointia tai atk-petosta vastaan, mutta näitä menetelmiä ei välttämättä ole suunniteltu suojaamaan varkauksilta, jotka kohdistuvat virtuaalisiin tuotteisiin. Vaikka lakien täytäntöönpanolla on tällaisia käytännön vaikeuksia, bitcoinin toiminnot on se tekijä, joka johtaa rikosoikeudellisten sääntöjen käyttöön liittyviin ongelmiin. [Boehm and Pesch 2014]

Ciaianin ja muiden [2016] mukaan menneisyydessä useat bitcoineja omistavat ovat menettäneet virtuaalirahojaan varkauden takia. Standardoidut valuutat antavat mahdollisuuden suojata itseään joko fyysisesti piilottamalla, esimerkiksi kassakaappiin, tai tallettamalla sitä pankkiin. Bitcoin on virtuaalinen valuutta, ja näin ollen sitä ei pysty fyysisesti piilottamaan, vaan bitcoineja pidetään virtuaalisissa lompakoissa. Juuri näiden virtuaalisten bitcoin-lompakoiden turvallisuus on usein ollut suuri ongelma [Ciaian *et al*. 2016]. Tämä johtuu lompakoiden sisältämän yksilöllisen avaimen varastamisesta. Esimerkiksi yksi virtuaalisten lompakoiden palveluista nimeltä mybitcoin.com hukkasi 1,3 miljoonan dollarin arvosta käyttäjien bitcoineja haittaohjelman takia [Barber *et al*. 2012].

Myös bitcoin-pörssit ovat olleet hyökkäyksien kohteina. Toisin kuin perinteiset valuutat, bitcoinit ovat olemassa ainoastaan virtuaalisena omaisuutena ja tilisiirtoina. Kun tilisiirto kerran suoritetaan, se on peruuttamaton. Tämä tekee bitcoinista hyvän kohteen hakkereille ja huijareille. Hienostuneet hyökkäykset bitcoin-pörsseihin ovat yleisiä [Ali *et al*. 2015]. Yksi esimerkki bitcoin-pörsseihin kohdistuneista iskuista oli Japanissa sijaitsevaan Mt. Gox -nimiseen pörssiin tehty hyökkäys, joka hyödynsi bitcoin-ohjelmistoa ja näin ollen salli samojen bitcoinien tuplakäytön pörssistä [Hurlburt and Bojanova 2014]. Iskusta merkittävän tekee se, että Mt. Gox oli iskuhetkellä maailman suurin bitcoin-pörssi, joka vastasi noin 80% kaikista bitcoin-tilisiirroista. Iskun seurauksena yli puolen miljardin dollarin arvosta käyttäjien bitcoineja varastettiin. Tämä vaikutti huomattavasti käyttäjien luottamukseen bitcoinia kohtaan, mikä näkyi bitcoinin arvon laskussa [Ali *et al*. 2015].

4.3.2 Rahanpesu, veropetokset ja laiton kaupankäynti

Osa bitcoin-tilisiirroista liittyy rikolliseen toimintaan [Ciaian *et al*. 2016]. On mahdollista vaihtaa bitcoineiksi raha, joka tulee laittomasta toiminnasta, ja sen jälkeen salata tämän rahan alkuperä uudestaan. Bitcoin-tilisiirtojen jäljitettävyys on monimutkaista ja näin ollen on erittäin haastavaa viranomaisille varmistaa bitcoinien alkuperä. Bitcoinien tilisiirrot ovat paljon vähemmän kontrolloituja kuin tavallisten pankkien [Boehm and Pesch 2014].

On olemassa ”pesupalveluita”, jotka hyväksyvät bitcoineja useista lähteistä, minkä jälkeen ne sekoitetaan siten, että linkki syöte- ja tulosteviitteen välillä katkeaa [Ali *et al*. 2015]. Jotkut erityiset tapahtumat ovat nostaneet epäilyksiä, että bitcoineja käytettäisiin laittomien rahojen pesemiseen veropetoksista. Tunnetuin esimerkki on bitcoin-pörssin suhteen todella nopea nousu juuri ennen pakollista pankin ulosmittausta Kyproksella maaliskuussa 2013. Silloin bitcoinin siirtosuhde tuplaantui muutamassa päivässä, eikä ole laskenut sen jälkeen [Boehm and Pesch 2014]. Boehmin ja Peschin [2014] mukaan yksi syy tähän nopeaan nousuun voisi olla pankin tilinomistajien yritys vaihtaa rahansa anonyymiin valuuttaan piilottaakseen rahan alkuperän ja suojatakseen sitä talousviranomaisilta [Boehm and Pesch 2014].

Alin ja muiden [2015] mukaan rahanpesu ei välttämättä ole välitön uhka bitcoinien rajatun käytön ja korkean hintaepävakaisuuden takia. Kuitenkin tutkijat ovat alkaneet kuulostella asiaa ja Yhdysvaltojen hallitus on asettanut rahanpesusääntöjä virtuaalisille valuutoille. Myös Europol on kehottanut laittamaan käytäntöön suurempia toimenpiteitä koskien tätä ongelmaa [Ali *et al*. 2015].

Bitcoin-tilisiirtojen anonymiteetti tekee bitcoinista houkuttelevan työkalun rikollisille, jotka käyttävät sitä laittomaan toimintaan. Kun verrataan bitcoinia tavalliseen rahaan, ovat bitcoinin edut kaksijakoisia: ei ole pakko olla henkilökohtaisesti läsnä, kun vastaanottaa rahaa, eikä ole pakko käyttää pankkitilejä, joita kontrolloidaan ja joilla voidaan mahdollistaa tunnistaminen. Bitcoin-tilisiirrot, jotka käsitellään bitcoin-pesupalvelussa, ovat paljon vaikeammin viranomaisten todennettavissa ja kontrolloitavissa kuin normaalien pankkitilien tapauksissa, vaikka bitcoinit kulkisivatkin välikäden kautta. Näiden tunnusmerkkien perusteella voidaan sanoa, että bitcoinien käytöstä on tulossa yhä suositumpaa laittomien tuotteiden ostamisessa anonyymeistä verkostoista. [Boehm and Pesch 2014]

Tunnetuin esimerkki tällaisesta verkostosta on Silk Road, jossa myytiin huumeita ja muita laittomia tuotteita vuodesta 2011 alkaen [Boehm and Pesch 2014]. Alin ja muiden [2015] mukaan verkoston tavoitteena oli, että myyjien identiteetti saataisiin suojattua täysin, kun taas ostajien täytyisi toimittaa myy-

jille fyysinen toimitusosoite tuotteelle. Jos viranomaiset soluttautuisivat verkostoon laittomuuksien myyjinä, he voisivat onnistua saamaan ostajien osoitteita selville, mutta soluttautujat eivät saisi myyjistä mitään hyödyllistä informaatiota [Ali *et al*. 2015]. Silk Road vastasi jopa joka toisesta bitcoin-tilisiirrosta ennen kuin FBI ajoi sen alas vuonna 2013. Usein on väitetty, että tämä tapahtuma lisäsi bitcoinin suosiota [Ciaian *et al*. 2016].

5. Bitcoinin edut

Tässä kappaleessa käsitellään bitcoinin etuja, joita ovat rahansiirron alhainen hinta, käyttäjän yksityisyys, turvallisuus sekä lohkoketju ideana. Bitcoinilla on todettu olevan myös muita etuja, mutta tässä tutkielmassa keskitytään edellä mainittuihin tapauksiin.

Yksi bitcoinin suurista eduista on rahansiirron alhainen hinta. Luottokorttien korkeat siirtomaksut verrattuna bitcoinin alhaisiin hintoihin tilisiirroissa on nähty tärkeänä argumenttina bitcoinin puolesta [Baur et al. 2015]. Baurin ja muiden [2015] mukaan alhaiset tilisiirtohinnat olivat tärkein yksittäinen tekijä elinkeinonharjoittajille maksutapoja vertailtaessa, myös mikromaksujen osalta. Bitcoin uskoo, että suurin osa solmuista verkostossa on rehellisiä, ja bitcoin turvautuukin siksi enemmistön äänestysmekanismiin välttääkseen bitcoinien tuplakäytön. Tämä palvelee hyvin ihmisiä, jotka uskovat vapaasti vaihdettavaan valuuttaan, joka ei ole minkään hallituksen, pankin tai viranomaisten hallinnassa [Barber et al. 2012].

Bitcoinin varmistajien markkinat perivät todella alhaisia tilisiirtomaksuja, jotka ovat vapaaehtoisia ja jotka maksaja saa päättää. Tämä voi olla houkuttelevaa mikromaksuissa, joissa tilisiirtomaksut saattavat dominoida. Bitcoin on myös siinä mielessä houkutteleva, että systeemistä puuttuu ylimääräiset maksut, jotka perinteisesti liitetään kansainvälisiin rahansiirtoihin [Barber *et al*. 2012]. Baurin ja muiden [2015] mukaan myös tilausten maksujen siirtyminen tileille välittömästi nähtiin hyödyllisenä asiana.

Kuluttajat arvostivat anonymiteettiä ja turvallisuutta todella paljon [Baur et al. 2015]. Baurin ja muiden [2015] mukaan käyttäjät pelkäsivät joutuvansa luottokorttihuijauksen uhriksi, ja he toivoivat, että bitcoin voisi olla lääke luottokorttihuijauksiin. Hurlburtin ja Bojanovan [2014] mukaan bitcoin toimii kuten käteinen, joka vähentää identiteettivarkauksia ja luottokorttihuijauksia. Bitcoin-tilisiirrot ovat peruuttamattomia, mikä houkuttelee kauppiaita, jotka ovat huolestuneita luottokorttihuijauksista ja takaisinveloituksista [Barber *et al*. 2012]. Barberin ja muiden [2012] mukaan eräs kauppias ei pystynyt tekemään bisnestä asiakkaiden kanssa tietyissä maissa, joissa luottokorttihuijaukset vallitsevat.

Bitcoinin avulla hän pystyy laajentamaan liiketoimintaansa näihin maihin tilisiirtojen peruuttamattomuudesta saadun suojan takia.

Bitcoinin perustana oleva lohkoketju ideana ja teknologiana nähdään potentiaalisena myös virtuaalimaksamisen ulkopuolella. Sovellusalueet kuten dokumenttien versionhallinta, todiste siitä, onko henkilön ääni laskettu tai selvä tunnistaminen verkkohallinnossa ja avoimen datan ratkaisuissa, oli ehdotusten joukossa. [Baur *et al*. 2015]

6. Nykytila ja tulevaisuus

Bitcoin on suurin virtuaalivaluutta tällä hetkellä. Sen markkina-arvo on noin 7,1 miljardia dollaria, kun toisiksi suurimmalla virtuaalivaluutalla nimeltä Ethereum markkina-arvo on noin 750 miljoonaa dollaria.

#	Name	Market Cap
1	Bitcoin	$ 7,090,481,540
2	Ethereum	$ 763,017,061
3	Ripple	$ 223,280,194
4	Litecoin	$ 178,720,735
5	Dash	$ 42,684,260

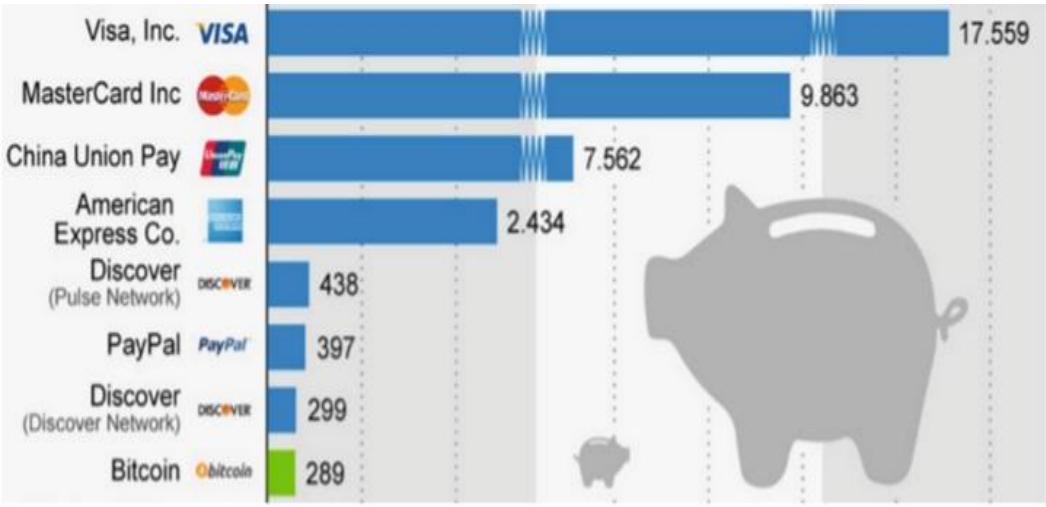
Kuva 3. Virtuaalivaluuttojen markkinaosuudet 9.5.2016 [Coinmarketcap Currencies]

Kuvassa 3 on viisi suurinta virtuaalivaluuttaa ja niiden markkina-arvot. Kuvasta huomataan, että bitcoin dominoi virtuaalivaluuttojen markkinoita, ja kaikista virtuaalivaluutoista bitcoineja onkin 82,2% [Coinmarketcap Currencies]. Muita virtuaalivaluuttoja kutsutaan vaihtoehtorahaksi (englanniksi altcoins).

Suomessa bitcoineilla voi maksaa 41 kivijalkakaupassa ja 44 verkkokaupassa. Verkkokaupat ovat suurimmalta osalta keskittyneet elektroniikkaan, mutta myös esimerkiksi armeijatavaran erikoisliike Varusteleka hyväksyy bitcoinit. Tampereella sijaitsevat Puls & Träning –kuntosalit hyväksyvät bitcoinit maksutapana. [Bittiraha Maksupaikat]

Kuvassa 4 on vertailtu bitcoinia muihin verkossa maksamisen menetelmiin. Huomataan, että Visa hallitsee tätä osa-aluetta, ja bitcoinilla on vielä tekemistä päästäkseen todelliseksi tekijäksi. Käyttäjien mielipide bitcoinista on kuitenkin

erittäin positiivinen; he mieltävät sen todella potentiaaliseksi. Kuitenkin luottokortit, PayPal ja jotkut muut maksupalvelut ovat nyt, ja vielä vähän aikaa tulevaisuudessa, suositumpia verkkomaksamisen menetelmiä kuin bitcoin. Koska bitcoinin asiakkaiden kysyntä on todella alhainen, ja bitcoin on vielä pelkkä erikoistuote pienillä markkinoilla, niin se ottaa aikansa kehittyä suuremmaksi [Baur *et al*. 2015].



Kuva 4. Päivittäinen rahansiirtojen volyymi miljoonissa dollareissa vuoden 2013 lopussa [Baur *et al*. 2015]

Bitcoinilla on suuri etu verrattuna tavalliseen valuuttaan maissa, joissa on epävakaa taloudellinen systeemi. Bitcoin saattaa antaa vaihtoehdon tavalliselle valuutalle maissa, joissa on köyhät, harvassa olevat ja kalliit taloudelliset palvelut, ei vaihtokelpoinen valuutta ja korkea hallinnollinen taakka avattaessa tiliä. Vaihtoehtoisesti bitcoin voisi edustaa kustannustehokasta maksupalvelua kehittyvissä maissa, joissa perinteiset rahansiirrot ovat todella kalliita ja pankkisysteemi on kehittymätön ja turvaton. Bitcoin-rahansiirrot voidaan tehdä suhteellisen pienillä maksuilla ja resurssivaatimuksilla, ja koska bitcoin on vapaa maantieteellisestä sijainnista ja pankkisysteemistä, se on ihanteellisesti sijoittautunut tarjoamaan tehokasta kansainvälistä maksusysteemiä. [Ciaian *et al*. 2016]

Lutherin [2015] mukaan pitkän aikavälin todennäköisyys ei suosi bitcoinia, mutta jos lohkoketju-teknologia pystyy huomattavasti alentamaan rahansiirtojen käsittelyn maksuja, se tullaan omaksumaan. Kuitenkin sekä Ciaian ja muut [2016], että Baur ja muut [2015] olivat sitä mieltä, että lohkoketju avaa useita uusia mahdollisuuksia ja teknologisia innovaatioita. Muun muassa turvallinen äänestyssysteemi, yleinen pörssi ja dokumenttien versionhallinta nähtiin tulevaisuudessa tapauksiksi, joissa lohkoketjusta olisi paljon hyötyä.

7. Yhteenveto

Bitcoinin perustana oleva teknologia on melko monimutkaista, ja tekniikan tai kaupan alasta tietämättömälle saattaa olla vaikeaa tajuta bitcoinin toimintaa ilman kunnon perehdytystä. Ciaianin ja muiden [2016] mukaan bitcoin on suhteellisen uusi valuutta ja sen hinnanmuodostumista ei ymmärretä vielä kunnolla, koska saatavana on vasta muutama bitcoinin hinnanmuodostumista koskeva kirjallinen tutkimus. Tavalliselta bitcoinin käyttäjältä vaaditaan siis suurta omatoimista perehtymistä asiaan, että sen ymmärtää kunnolla.

Bitcoinin perustat eli louhinta ja käyttäjän anonymiteetti ovat sekä bitcoinin parhaita että ongelmallisia puolia. Tässä tutkielmassa käsitellyt ongelmat eivät kuitenkaan ole suurempia kuin niistä saatava hyöty, ja näin tuoreella keksinnöllä on varmasti ongelmia myös tulevaisuudessa. Bitcoinin edut liittyvät juuri näihin asioihin, joista samalla muodostuvat bitcoinin perustat ja ongelmat. Näyttää siis siltä, että bitcoinin ongelmat ovat lähtöisin sen parhaista puolista. Tämä varmaankin johtuu siitä, että näitä kohtia on tutkittu eniten ja samalla niitä halutaan varjella ja kehittää.

Viiteluettelo

Syed Taha Ali, Dylan Clarke, and Patrick McCorry. 2015. Bitcoin: Perils of an Unregulated Global P2P Currency. In: *Revised Selected Papers of the 23rd International Workshop on Security Protocols, Lecture Notes in Computer Science* 9379. Springer, 283-293.

Elli Androulaki, Ghassan O. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating User Privacy in Bitcoin. In: *Revised Selected Papers of the 17th International Conference on Financial Cryptography and Data Security (FC 2013), Lecture Notes in Computer Science* 7859. Springer, 34-51.

Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. 2012. Bitter to Better – How to Make Bitcoin a Better Currency. In: *Revised Selected Papers of the 16th International Conference on Financial Cryptography and Data Security (FC 2012), Lecture Notes on Computer Science* 7397. Springer, 399-414.

Aaron W. Baur, Julian Bühler, Markus Bick, and Charlotte S. Bonorden. 2015. Cryptocurrencies as a Disruption? Empirical Findings on User Adoption and Future Potential of Bitcoin and Co. In: *Proc. of the 14th International Conference on Open and Big Data Management and Innovation, Lecture Notes in Computer Science* 9373. Springer, 63-80.

Michael Bedford. 2013. Bitcoin and The Age of Bespoke Silicon. In: *Proc. of the 2013 International Conference on Compilers, Architectures and Synthesis for Embedded Systems (Cases ´13)*. IEEE, Article No. 16.

[Bitcoin Address] Bitcoin Wiki, Address. Available at https://en.bitcoin.it/wiki/Address. Checked 19.04.2016.

[Bitcoin ASIC] Bitcoin Wiki, ASIC. Available at https://en.bitcoin.it/wiki/ASIC. Checked 29.3.2016.

[Bitcoin Block hashing algorithm] Bitcoin Wiki, Block hashing algorithm. Available at https://en.bitcoin.it/wiki/Block_hashing_algorithm. Checked 14.3.2016.

[Bitcoin FPGA] Bitcoin Wiki, FPGA. Available at https://en.bitcoin.it/wiki/FPGA. Checked 29.3.2016.

[Bitcoin Mining hardware comparison] Bitcoin Wiki, Mining hardware comparison. Available at https://en.bitcoin.it/wiki/Mining_hardware_comparison. Checked 1.4.2016.

[Bitcoin Non-specialized hardware comparison] Bitcoin Wiki, Non-specialized hardware comparison. Available at https://en.bitcoin.it/wiki/Non-specialized_hardware_comparison. Checked 31.3.2016.

[Bitcoin Nonce] Bitcoin Wiki, Nonce. Available at https://en.bitcoin.it/wiki/Nonce. Checked 12.3.2016.

[Bitcoin Target] Bitcoin Wiki, Target. Available at https://en.bitcoin.it/wiki/ Target. Checked 12.3.2016.

[Bittiraha Maksupaikat] Bittiraha, Maksupaikat. Available at https://bittiraha.fi/content/bitcoin-karttapalvelu. Luettu 9.5.2016.

Franziska Boehm, and Paulina Pesch. 2014. Bitcoin: A First Legal Analysis. In: *Revised Selected Papers of the workshop on Financial Cryptography and Data Security (FC 2014), Lecture Notes on Computer Science* 8438. Springer, 43-54.

Pavel Ciaian, Miroslava Rajcaniova, and d'Artis Kancs. 2016. The digital agenda of virtual currencies: Can BitCoin become a global currency?. to appear in *Inf Syst E-Bus Manage.*

[Coinmarketcap Currencies] Coinmarketcap, Currencies. Available at http://coinmarketcap.com/currencies/. Checked 9.5.2016.

Ittay Eyal, and Emin Gün Sirer. 2014. Majority Is Not Enough: Bitcoin Mining Is Vulnerable. In: *Revised Selected Papers of the 18th International Conference on Financial Cryptography and Data Security (FC 2014), Lecture Notes in Computer Science* 8437. Springer, 436-454.

George F. Hurlburt, and Irena Bojanova. 2014. Bitcoin: Benefit or Curse?. *IT Professional* 16, 3, 10-15.

William J. Luther. 2015. Bitcoin and the Future of Digital Payments. *Available at Social Science Research Network*.

Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. *www.bitcoin.org.*

Karl J. O'Dwyer, and David Malone. 2014. Bitcoin Mining and its Energy Footprint. In: *Proc. of the Irish Signals & Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies (ISSC 2014/CIICT 2014).* 25th IET. 280-285.

Aviv Zohar. 2015. Bitcoin: Under the Hood. *Comm. of the ACM* 58, 9, 104-113.

Pelillistäminen ohjelmoinnin opetuksessa

Juha Kauppila

Tiivistelmä.
Ohjelmoinnin opetus lisääntyy tulevaisuudessa mm. Suomessa, kun sitä halutaan opettaa yhä isommalle joukolle oppilaita. Opetuksen menetelmien kehittäminen on siis ajankohtaista ja pelillistäminen suhteellisen tuoreena ilmiönä tarjoaa yhden tavan opiskelijan auttamiseen ja motivoimiseen. Tässä tutkielmassa tutustutaan esimerkkitapausten kautta, miten ohjelmoinnin opetuksen pelillistämistä on pyritty tekemään ja millaisia tuloksia nämä ovat saavuttaneet. Pelillistämisen suurimmat hyödyt esimerkkitapausten valossa ovat sitoutumisen ja motivaation lisääntyminen.

Avainsanat ja -sanonnat: Pelillistäminen, ohjelmoinnin opetus, tapaustutkimukset, pelimekaniikat.

1. Johdanto

Julkistettujen tutkimusten tuloksiin pohjaten Watson ja Li [2014] laskivat, että maailmanlaajuisesti keskimäärin joka kolmas oppilas ei läpäise ohjelmoinnin perusteiden kurssia. Ongelmat ohjelmoinnin perusteiden opettamisessa ovat Suomessakin varsin ajankohtaisia, sillä peruskoulujen opetussuunnitelma sisältää syksystä 2016 alkaen ohjelmoinnin opetusta kaikille peruskoululaisille [Mykkänen ja Liukas 2014].

Pelillistäminen (gamification) on yksi mahdollisista tavoista, joilla oppilaita voitaisiin avustaa ohjelmoinnin opiskelussa. Tässä tutkielmassa käydään läpi kirjallisuudessa käsiteltyjä pelillistettyjä ohjelmointikursseja tai oppimisympäristöjä ja niiden tuottamia tuloksia. Pelillistäminen terminä ja menetelmänä on suhteellisen uusi. Deterdingin ja muiden [2011] mukaan ensimmäinen maininta tutkimuskirjallisuudesta löytyy vuodelta 2008, mutta termin käyttö yleistyi vasta vuoden 2010 loppupuolella.

Pelillistämistä ja siihen liittyviä käsitteitä käsitellään luvussa 2. Tässä tutkielmassa keskitytään pääasiassa pelillistämiseen, mutta luvussa 2 esitellään myös vaihtoehtoisia tai kilpailevia termejä ja tutustutaan lyhyesti pelillistämiseen kohdistuvaan kritiikkiin. Kolmannessa luvussa esitellään tarkemmin kirjallisuushaussa löydetyt tutkimukset ja tarkastellaan, millaisia pelillistämisen mekanismeja ne sisältävät. Luvussa 4 käydään läpi kolmannessa luvussa esiteltyjen tutkimusten tuloksia ja analysoidaan niitä. Tuloksia vertaillaan myös Vihavaisen ja muiden [2014] esittelemiin tuloksiin.

2. Määritelmistä ja mekanismeista

Deterding ja muut [2011] määrittelevät pelillistäminen ”pelisuunnittelun elementtien käyttämiseksi ei-pelillisissä yhteyksissä”. Määritelmä ei rajaa pois ei-digitaalisia käyttökohteita, mutta tämän työn yhteydessä keskitytään joitain poikkeuksia lukuun ottamatta digitaalisiin sovelluksiin. Pelillisten elementtien lisäämisen perimmäinen tarkoitus on sitouttaa ja motivoida käyttäjää sovelluksen käyttämiseen ja toisaalta käyttäjän näkökulmasta tehdä mahdollisesti tylsästä tai työläästä asiasta hauskempaa.

Tutkimuksesta löytyy useita kilpailevia tai rinnakkaisia termejä pelillistämisen ilmiölle. Deterdingin ja muiden [2011] mukaan muita käsitteitä ovat esimerkiksi *leikillinen suunnittelu* (playful design), *tuottavuuspelit* (productivity games) ja *pelikerros* (game layer). Päästäksemme kritiikkiin on syytä esitellä tarkemmin yksi vaihtoehtoinen termi *pelimäinen suunnittelu* (gameful design). Pelimäinen suunnittelu voidaan määrittää pääpiirteissään samalla tavalla kuin pelillistäminen, mutta Deterding ja muut [2011] hahmottavat niiden eroavaisuutta tavoitteiden kautta. Pelillistämisen tavoite on pelien elementtien käyttäminen muussa sovelluksessa kuin pelissä, kun taas pelimäisen suunnittelun päämääränä on luoda sovelluksen käyttämisen kokemuksesta mahdollisimman pelimäinen.

2.1. Pelillistämisen tekniikoita

Pelillistämisen elementtejä tai tekniikoita ovat Zichermannin ja Cunninghamin [2011] mukaan *pisteet* (points), *kokemustasot* (levels), *pistetilastot* (leaderboards), *kunniamerkit* (achievements, badges), *haastetehtävät* (challenges), *perehdyttäminen* (onboarding) ja *sitouttamissilmukat* (feedback loops). Karkeasti ottaen näitä elementtejä käytetään joko palkitsemiseen (pisteet, kokemustasot, kunniamerkit), käyttäjän kilpailuhengen herättelyyn (haastetehtävät, pistetilastot) tai sitouttamiseen (perehdyttäminen, sitouttamissilmukat).

Fui-Hoon Nah ja muut [2014] listasivat kirjallisuustutkimuksessaan kahdeksan elementtiä, joita opetuskäyttöön tarkoitetuissa ympäristöissä oli enimmäkseen käytetty. Näitä olivat pisteet, kokemustasot, kunniamerkit, pistetilastot, *palkinnot* (prizes, rewards), *edistymispalkit* (progress bars), *tarinallisuus* (storyline) ja *palaute* (feedback).

Pisteet. Pisteitä käytetään usein saavutusten tai menestyksen mittarina. Joskus näitä pisteitä voi myös käyttää pelin sisäisesti muiden etujen tai palkintojen saavuttamiseen. Zichermann ja Cunningham [2011] esittävät, että pisteet ovat vaatimus pelillistetylle systeemille, vaikka niitä ei käyttäjille näytettäisikään. Heidän mukaansa pisteiden avulla on helppo seurata käyttäjän toimintaa ohjelmassa.

Kokemustasot. Kokemustasoja käytetään indikoimaan käyttäjälle hänen etenemistään. Alussa tasoja saavuttaa usein helpommin, mutta niiden saaminen käy haastavammaksi mitä pidemmälle etenee. Kokemustasot voivat olla sidoksissa pistemäärään tai olla siitä erillinen systeemi. Fui-Hoon Nah ja muut [2014] havainnoivat, että vaikka kokemustasot ovat yleinen pelillistämisen elementti, opiskelijoiden taidot eivät välttämättä parane tai kehity tasojen mukana.

Pistetilastot. Pistetilastojen tavoite on motivoida käyttäjää ja herättää kilpailua käyttäjien kesken. Usein pistetilastot näyttävät vain osan käyttäjistä tilastojen kärkipäästä, etteivät huonoilla sijoituksilla olevat menetä motivaatiotaan. Pistetilastot ovat yleensä yhteydessä pisteisiin, mutta voisivat olla esimerkiksi myös kokemustasoihin tai kunniamerkkeihin pohjautuvia.

Kunniamerkit. Kunniamerkit ovat yleensä palkintoja käyttäjän toimista tai saavutuksista. Kunniamerkkien tehtävä on motivoida käyttäjiä kohti tiettyjä tavoitteita tai jatkamaan tehtävien tekemistä, vaikka minimivaatimus olisikin jo saavutettu. Kunniamerkkejä voi myös usein jakaa käyttäjien kesken tai sosiaalisessa mediassa.

Haastetehtävät. Haastetehtävät tarjoavat käyttäjille tavoitteita, joita kohti työskennellä. Opetusympäristössä haastetehtävät voivat tarjota käyttäjälle myös lisää tekemistä sen jälkeen, kun kurssin tavoitteet on suoritettu. Haastetehtävien suorittamisesta voi saada palkinnoksi pisteitä tai kunniamerkkejä.

Perehdyttäminen. Perehdyttäminen tarkoittaa yksinkertaisesti haasteen räätälöimistä niin, että käyttäjä ei ahdistu törmätessään liian vaativiin tehtäviin pelillistetyssä ympäristössä. Perehdyttäminen ohjaa käyttäjän hitaasti kohti suurempia haasteita niin, että käyttäjän kyvyt kasvavat haasteen mukana.

Sitouttamissilmukat. Zichermann ja Cunningham [2011] jakavat sitouttamissilmukan neljään osaan: motivaatioon käyttää sovellusta, pelaajan uudelleensitouttamiseen, yhteisön tapahtumiin osallistumiseen ja palkitsemiseen. Näiden tarkoitus on sitouttaa käyttäjä peliin ja saada hänet osallistumaan yhä uudestaan sen käyttämiseen.

Palkinnot. Palkinnoilla tarkoitetaan muita palkitsemisen välineitä kuin pisteet, kokemustasot ja kunniamerkit. Fui-Hoon Nah ja muut [2014] käyttävät esimerkkinä pelaajan hahmon ulkonäön, ominaisuuksien tai varusteiden päivityksiä etenemisen palkintona. Tämä luonnollisesti vaatii, että pelillistetty ympäristö sisältää pelihahmoja.

Edistymispalkit. Edistymispalkkien ero kokemustasoihin ja kunniamerkkeihin on, että siinä missä kokemustasot ja kunniamerkit kertovat jo saavutetuista tavoitteista, edistymispalkin avulla voidaan havainnollistaa sitä, miten lähellä tai kaukana tavoitteen saavuttaminen on. Fui-Hoon Nah ja muut [2014] esittävät

myös, että edistymispalkkien avulla voitaisiin rohkaista opiskelijoita jatkamaan, jos he ovat jäämässä jälkeen tavoitteiden saavuttamisessa.

Tarinallisuus. Tarinallisuudella viitataan pelillistetyn ympäristön kerrontaan tai tarinaan. Tarinoilla voidaan tarjota oppimiselle jokin viitekehys tai kerrontaa voi käyttää havainnollistamaan miten opetetut asiat liittyvät tai miten niitä voi käyttää oikeissa tilanteissa.

Palaute. Palaute on tärkeä sitouttamisen väline. Palaute voi liittyä muihin mekaniikkoihin kuten pisteet tai kunniamerkit, mutta voi olla jotain muutakin. Fui-Hoon Nah ja muut [2014] kirjoittavat, että selkeä ja välitön palaute on tärkeä tekijä flow-tilan saavuttamisessa. Flow-tila puolestaan edesauttaa sitoutumista.

2.2. Pelillistämisen kritiikistä

McGonicalin [2011] mukaan pelillistämisen neljä suurta ideaa ovat pisteet, kokemustasot, pistetaulukot ja kunniamerkit. Vastaavasti McGonicalin [2011] listauksessa pelimäisen suunnittelun neljä suurta ideaa ovat *positiivisuus* (positive emotion), *suhteet* (relationships), *merkitys* (meaning) ja *saavutukset* (accomplisment). McGonical [2011] näkee pelillistämisen ideat ulkoisena palkitsemisena, kun taas pelimäinen suunnittelu pyrkii palkitsemaan sisäisesti. Tämä on hänen kritiikkinsä ydin; pelaaminen ja pelit ovat muutakin kuin mitä pelillistämisellä on tähän mennessä pinnallisesti yritetty tehdä.

McGonical ei ole ainoa kriitikko, joka on kiinnittänyt huomiota pelillistämisen toteutuksen epäkohtiin. Bogost [2011] kirjoittaa kolumnissaan kärkevään sävyyn pelillistämisestä ja ehdottaakin, että pelillistämisestä puhumisen sijaan ilmiötä pitäisi kutsua *riistosovelluksiksi* (exploitationware). Hänen argumenttinsa on, että pelillistäminen-termin käyttö on vain retorinen keino piilottaa kaikki ne vaikeudet, jotka pelin tekemiseen liittyy. Saati sitten mitä ongelmia johonkin muuhun tarkoitukseen tehdyn sovelluksen pelimäistämiseen kuuluu. Bogost [2011] toteaa kolumnissaan, että pelien tekeminen on vaikeaa, hyvien pelien tekeminen sitä vaikeampaa ja sellaisten hyvien pelien tekeminen, joilla on jokin pelin ulkoinen tavoite, on vieläkin vaikeampaa.

Bogost [2011] esittää pelillistämisen nykymuodossaan palvelevan vain markkinointia ja konsultteja, jotka pystyvät myymään määritellyn kaltaista pelillistämistä helposti tuotteena. Bogostin [2011] mukaan ”lisää vain pisteet, kunniamerkit ja pistetilasto”-pelillistäminen erehtyy pitämään mainittuja elementtejä pelien ensisijaisena tunnusmerkkinä, vaikka ne ovat vain toissijainen lisä pelien vetovoimaisuudessa.

3. Tutkimuslähteiden esittely

Tässä luvussa käydään läpi lähteenä käytetyissä tutkimuksissa esiintyneet ohjelmoinnin opetukseen liittyvät pelillistetyt ratkaisut. Ratkaisut kuvaillaan lyhyesti ja esitellään millaisia pelillistämisen menetelmiä niissä on käytetty. Taulukossa 1 on yhteenveto käsitellyistä tutkimuksista ja niissä käytetyistä pelillistämismenetelmistä.

	Käytetyt pelillistämismenetelmät	**Viite**
Extreme programming	pisteet, pistetilastot, kunniamerkit, haastetehtävät, sosiaalinen dynamiikka	[Akpolatin and Slany 2014]
Redmine	pisteet, pistetilastot	[Buismanin and van Eekelen 2014]
Kunniamerkit ja A+	kunniamerkit	[Haaranen *et al.* 2014]
Q-Learning-G	pisteet, kunniamerkit	[Ibáñezin *et al.* 2014]
Javala	pisteet, haastetehtävät, tasot, pistetilastot	[Lehtonen *et al.* 2014]
Khan Academy	kunniamerkit, pisteet, tavoitteet, edistymispalkit	[Morrison and DiSalvo 2014]
Order of the Curmudgeons	tarinallisuus, palkinnot, pisteet, edistymispalkit, kunniamerkit, pistetilastot	[O'Donovan *et al.* 2013]

Taulukko 1. Lähteissä käytetyt pelillistämismenetelmät

3.1. Extreme programming -kurssi

Akpolatin ja Slanyn [2014] tutkimuksessa pelillistämisen kohteena on ollut vuonna 2013 järjestetty Graz University of Technologyn extreme programming (XP) -kurssi. Tutkimukseen osallistujia oli 50 ja heidät oli jaettu viiteen projektiryhmään, joissa kaikille oli jaettu omat roolinsa (kehittäjä, asiakas, käytettävyysasiantuntija, jne.). Joka viikko osallistujat tekivät työpäivän verran (8 tuntia) kurssitöitä. Kurssin kesto oli 14 viikkoa, joista tutkimuksessa seurattiin vain 6 ensimmäistä viikkoa.

Jokaiselle viikolle oli määritelty jokin XP:n käytäntöihin ja niiden toteuttamiseen liittyvä haaste. Haasteen sujumista ja pisteytystä arvioi tutkimuksen järjestäjä. Viikon päätteeksi yksi ryhmistä valittiin viikon voittajaksi ja tutkimuksen lopussa eniten voittoja saanut ryhmä sai palkinnon. Ensimmäisen viikon jälkeen arvioitiin myös edellisillä viikoilla käsiteltyjen XP:n käytäntöjen toteutumista, mutta tästä ei kerrottu tutkimukseen osallistujille.

Haasteiden lisäksi ryhmien toimintaa seurattiin kyselyillä. Kerran viikossa satunnaiselta ryhmän jäseneltä kysyttiin hänen arvioitaan siitä, miten hyvin ryhmä ymmärsi viikon XP-käytännön. Kurssin aikana kaikki osallistujat vastasivat lisäksi kahteen kyselyyn, jotka järjestettiin tutkimuksen puolivälissä ja tutkimuksen lopuksi.

Tutkimuksessa käytetyt pelillistämisen menetelmät sisältävät pisteet (ryhmien toiminta pisteytettiin viikottain), pistetilastot (ryhmät pystyivät seuraamaan pistetilannetta verkossa), kunniamerkit (viikkohaasteen voittanut ryhmä sai kunniamerkin pistetilastoon), haastetehtävät (viikottainen haaste) ja sosiaalisen dynamiikan hyödyntämisen (kilpailu oli ryhmien, ei yksilöiden, välistä).

3.2. Redmine

Redmine on Ruby on Rails-ohjelmistokehyksellä rakennettu avoimen lähdekoodin projektienhallintatyökalu, jota Alankomaissa sijaitsevan Radboud Universityn opiskelijavetoinen ohjelmistoyritys GiPHouse käyttää. Redminen avulla GiPHouse pyörittää asiakasprojekteja, seuraa niiden etenemistä, projektiin osallistujien ajankäyttöä ja ylläpitää projektien dokumentaatioita. GiPHousen työvoimana toimii yliopiston ohjelmistoprojektikursseille osallistuvat opiskelijat.

Buismanin ja van Eekelenin [2014] tavoitteena oli selvittää, miten pelillistäminen vaikuttaa Redminen käyttöön ja oppilaiden motivaatioon. Tutkimusta varten Redmineen lisättiin liitännäinen, joka laski pisteitä ohjelmistossa tehdyille toiminnoille ja piti yllä pistetilastoa osallistujista. Ohjelman toimintojen pisteyttämisestä vastasivat kurssien opettajat sen mukaan, mitä projekti tai GiPHouse milloinkin eniten tarvitsi. Pistetilasto oli anonymisoitu niin, että osallistuja näki vain oman kokonaissijoituksensa ja kahden ylä- ja alapuolellaan olevan pisteet. Lisäksi pistetilasto näytti, miltä osa-alueelta pisteet olivat kertyneet ja vertailun vuoksi sen, miltä osa-alueilta koko osallistujakunta keskimäärin oli saanut pisteitä.

Tutkimuksen aikana seurattiin kolmea ryhmää. Kontrolliryhmällä pelillistettyjä elementtejä ei ollut näkyvissä ollenkaan. Toisella ryhmällä oli näkyvissä vain heidän omat pisteensä, eivätkä he voineet verrata pisteitään muiden pisteisiin. Kolmannella ryhmällä näkyivät heidän omat pisteensä ja he pystyivät vertaamaan pisteitään muiden pisteisiin.

Tutkimuksessa seurattiin Redminen käyttöä kuuden kuukauden ajan. Tänä aikana Redminessä tehdyistä toiminnoista ja pisteistä kerättiin tilastodataa. Tutkimusajan päätyttyä osallistujille lähetettiin kysely, jonka kautta pyrittiin selvittämään sitoutumista, osallistumista ja motivaatiota. Osallistuneiden opiskelijoiden kurssiarvosanat yhdistettiin pisteisiin ja kyselyn tuloksiin analyysia varten.

Tutkimuksessa pelillistämiseen käytettiin pelkästään pisteitä ja pistetilastoja.

3.3. Kunniamerkit ja A+

Haaranen ja muut [2014] tutkivat Aalto-yliopiston pelillistetyllä tietorakenteiden ja algoritmien kurssilla, miten kunniamerkit vaikuttivat opiskelijoiden käytökseen. A+ on verkkosovellus, jossa opiskelijat voivat suorittaa kurssiin liittyviä tehtäviä. Tutkimusta varten A+-sovellukseen kehitettiin liitännäinen, joka tarkkaili opiskelijoiden toimintaa ja jakoi kunniamerkkejä sekä tarjosi näkymän käyttäjän ansaitsemiin kunniamerkkeihin.

Kurssin osallistujat jaettiin kahteen ryhmään, joista toinen sai suorituksista kunniamerkkejä ja toiselle ryhmälle ne eivät näkyneet mitenkään. Kurssin läpäisyn kannalta riitti, että opiskelija suoritti harjoitustehtävistä neljä ensimmäistä täysin pistein. Kunniamerkit näytettiin kuitenkin vasta viidennestä tehtävästä alkaen, eivätkä kunniamerkit vaikuttaneet kurssin suoritukseen, vaan olivat pelkästään visuaalinen lisä. Tehtäviin liittyviä kunniamerkkejä oli kolmea eri tasoa (pronssi, hopea ja kulta), ja niitä jaettiin kolmesta kategoriasta, oppimisesta, ajankäytöstä ja huolellisuudesta.

Kurssin jälkeen kaikille oppilaille lähetettiin anonyymi kysely, minkä lisäksi tutkijoiden käytössä oli tilastot oppilaiden suorituksista, suoritusten lähetysajoista ja suorituksista saadusta palautteesta sekä tieto siitä, milloin opiskelija oli käynyt katsomassa sivua, joka listasi hänen ansaitsemansa kunniamerkit.

3.4. Q-Learning-G

Ibáñez ja muut [2014] tutkivat pelillistämisen vaikutusta *sitoutumiseen* (engagement) ja C-ohjelmointikielen oppimiseen. Q-Learning-G-ohjelmisto kehitettiin tätä tutkimusta varten. Tutkimus suoritettiin Universidad Carlos III de Madrid -yliopistossa käyttöjärjestelmien kurssilla, osallistujia oli 22.

Opiskelijoiden tavoitteena oli saada kurssin aikana Q-Learning-G-ohjelmassa 100 suorituspistettä. C-ohjelmointia koskevat monivalintatehtävät oli jaettu kymmeneen osa-alueeseen ja opiskelijan tuli ansaita jokaisesta osa-alueesta 10 pistettä vastaamalla tehtäviin. Suorituspisteiden keräämisen lisäksi opiskelijat laativat monivalintatehtäviä muille tai arvioivat muiden tekemiä tehtäviä ohjelmassa olevan arviointilomakkeen avulla. Tehtävien laatimisesta tai arvi-

oinnista saadut pisteet seurasivat kysynnän ja tarjonnan lakia. Jos osa-alueelle oli tehty paljon kyselyitä, osa-alueen kyselyiden arvioinnin pistearvo kasvoi ja päinvastoin jos kyselyitä oli vähän, niiden laatimisen pistearvo nousi. Pisteiden lisäksi monivalintatehtävien tekemisestä, arvioinnista ja niihin vastaamisesta oli tarjolla kunniamerkkejä.

Tutkimusta varten oppilaiden toimet Q-Learning-G-ohjelmassa tilastoitiin ja heille lähetettiin tutkimuksen aikana viisi kyselyä, joilla kerättiin dataa opiskelijoista ja heidän mielipiteistään ja suoriutumisestaan. Opiskelijoiden C-kielen osaamista arvioitiin järjestämällä testi ennen ja jälkeen kurssin.

3.5. Javala

Javala oli avoin ja ilmainen verkkosovellus, jonka tarkoitus oli auttaa opiskelijoita siirtymään C++-kielestä Java-kieleen. Javalaa käyttäneessä oppilaitoksessa aloituskurssien opetuskielenä oli C++ ja myöhemmillä kursseilla Java. Lehtonen ja muut [2014] tarkastelevat Javala-ohjelman käyttöä sen yhdeksän toimintavuoden (2004-2013) aikana ja erityisesti pelillistämisen ja lokalisaation vaikutusta sen käyttöön.

Javala sisältää oppimateriaalia Java-kielen opetteluun ja teoriaosuuden päätteeksi tarjolla on tehtäviä, jotka käännetään ja arvioidaan automaattisesti. Onnistuneesti ratkaistusta tehtävästä annetaan pisteitä. Pisteiden ja ratkaistujen tehtävien kertyessä käyttäjä nousi tasoja (kunniamerkkejä) ”Javaturistista” ”Javakuninkaaksi” asti. Käyttäjien saavutukset nostettiin näkyviin muille ilmoituksissa ja nimimerkkien pisteet löytyivät päivän, viikon ja kaikkien aikojen top 100 -listoilla.

Tutkimuksen tietolähteenä on ollut käyttäjien statistiikka sovelluksen koko käyttöajalta ja vuoden 2013 alussa tehty kokeilu, jossa pelillistämiselementit poistettiin käytöstä kokonaan ja palautettiin sen jälkeen takaisin.

3.6. Khan Academy

Khan Academy on verkkosivusto, joka sisältää yli 4500 videokurssia [Morrison and DiSalvo 2014] aiheinaan matematiikkaa, luonnontieteitä, taloustieteitä ja ohjelmointia. Ohjelmoinnin opetuksessa käytössä on videoiden lisäksi interaktiivinen editori ja syötetyn koodin reaaliaikainen käännös ja arviointi. Vuonna 2010 sivusto pelillistettiin lisäämällä kursseihin kunniamerkkejä ja pisteitä.

Morrison ja DiSalvo [2014] tarkastelevat miten onnistuneesti Khan Academy on onnistunut pelillistämään sivustonsa. Morrison ja DiSalvo [2014] listaavat sivustolla käytetyiksi menetelmiksi kunniamerkit ja pisteet, tavoitteet, *edistymisindikaattorit* (progress indicator) ja kurssikartan. Kunniamerkkejä ja pisteitä voi saada mm. videoiden katselusta ja kurssien suorituksista. Kunniamerkit näkyvät sivustolla ja ne voi jakaa esimerkiksi Facebookin kautta. Kursseista

riippuen sivusto saattaa tarjota tavoitteita ja käyttäjä voi myös itse asettaa niitä itse itselleen. Edistymistä voi seurata useammallakin eri indikaattorilla. Tavoitteiden saavuttamista voi seurata edistymispalkista, joka näyttää, miten monta tavoitetta on saavutettu ja miten monta on saavuttamatta. Käyttäjän aktiivisuutta näytetään mm. siten, että listataan, miten paljon pisteitä tietyllä aikavälillä on saavutettu ja miten aktiivinen käyttäjä on ollut päivittäin.

3.7. "Order of the Curmudgeons"

O'Donovan ja muut [2013] halusivat selvittää, miten pelillistäminen vaikuttaa opiskelijoiden suoriutumiseen yliopistokurssilla. Pelillistämisen kohteena oli 2D-pelien suunnittelua ja kehitystä opettava kurssi Kapkaupungin yliopistossa vuonna 2012. Kyseinen kurssi oli valittu, koska pelit ja pelaaminen ovat ennestään tuttuja kurssin opiskelijoille ja toisaalta opiskelijat pystyivät miettimään pelillistämiseen käytettyjä menetelmiä kurssin opetussisällön pohjalta. Tavoitteena oli lisätä oppilaiden sitoutumista ja motivaatiota, parantaa luentojen kävijämääriä ja auttaa opiskelijoita kurssin sisällön ymmärtämisessä.

Ennen pelillistämisen toteutusta tutkijat lähettivät kurssin tuleville oppilaille kyselyjä, joilla selviteltiin, minkä tyyppisiä pelaajia osallistujat ovat ja millaisia pelillistämismekaniikkoja he pitivät hyödyllisimpinä. Näiden vastausten avulla tutkijat suunnittelivat pelillistetyn sisällön. Kapkaupungin yliopistossa on käytössä Vula-niminen verkko-oppimisympäristö, jonka päälle O'Donovan ja muut [2013] lisäsivät pelillisiä elementtejä. Kurssille oli suunniteltu taustatarina, jossa osallistujat liittyivät Äkämysten kiltaan (Order of the Curmudgeons) ja keräsivät vihjeitä selvittääkseen varastetun "Crowther Enginen" mysteeriä. Tutkijat muokkasivat Vula-ympäristön standardikurssipohjaa steampunk-teemaisella kuvituksella, jotta se sopi paremmin taustatarinaan.

Opiskelijat keräsivät kokemuspisteitä vastaamalla kurssin luentomateriaaleja koskeviin kyselyihin verkkoympäristössä viikoittain. Kyselyistä sai menestyksen mukaan kokemuspisteitä ja niitä sai yrittää kolme kertaa ennen kuin ne lukittuivat. Kyselyiden lisäksi verkkoympäristössä oli pulmatehtäviä, jotka eivät suoraan liittyneet luentomateriaaleihin, vaan testasivat opiskelijan ongelmanratkaisukykyä, näiden ratkaisusta palkintona oli kokemuspisteitä ja uusia vihjeitä tarinan mysteerin ratkaisuun. Kokemuspisteitä jaettiin myös toisinaan järjestetyistä ryhmätehtävistä. Tärkeänä osana oli myös luennoille osallistuminen ja luentoaktiivisuus, joista jaettiin myös kokemuspisteitä. Kurssilla järjestettiin normaalisti kokeita ja tehtäviä, joista ei saanut suoraan kokemuspisteitä, mutta kurssin järjestäjä saattoi jakaa ylimääräisiä kokemuspisteitä luovasta ohjelmoinnista tai suunnittelusta niiden yhteydessä.

Kerätyillä kokemuspisteillä sai pelin sisäistä valuuttaa, jolla pystyi ostamaan mm. uusia yrityksiä kyselyihin tai lykkäystä kurssin tehtäväpalautuksiin.

Kokemuspisteiden kokonaismäärän perusteella käyttäjä sai myös kunniamerkkejä ja hänen kokemustasonsa nousi. Kurssille osallistujat listattiin pistetilastossa, ja tilaston kymmenen parasta palkittiin kurssin lopussa t-paidalla. Lopuksi kokemuspisteillä oli myös pieni vaikutus kurssin arvosanaan.

Kurssin lopussa opiskelijoille lähetettiin kyselylomake, jolla selviteltiin heidän tuntemuksiaan siitä, miten pelillistäminen auttoi sisällön ymmärrystä, kurssiin sitoutumista ja heidän saamaansa arvosanaa. Tämän lisäksi tutkijat vertasivat saatuja arvosanoja edellisen vuoden vastaavaan kurssiin ja vertasivat luennoilla käyntimäärää muihin yliopiston kursseihin.

4. Tulosten analyysi

Tässä luvussa käsitellään luvussa 3 esiteltyjen tutkimusten tuloksia. Näiden tutkimusten saavuttamia tuloksia verrataan Hamarin ja muiden [2014] kokoamiin tuloksiin sekä Vihavaisen ja muiden [2014] samaa aihepiiriä käsittelevään tutkimukseen. Taulukossa 2 on yhteenveto lähteenä käytettyjen tutkimusten tuloksista.

	Tulokset
Extreme programming -kurssi	Lisääntynyt sitoutuminen, paremmat oppimistulokset
Redmine	Pelillistetyn ympäristön lisääntynyt käyttö
Kunniamerkit ja A+	Ei merkittäviä tuloksia
Q-Learning-G	Sitoutumisen ja motivaation lisääntyminen
Javala	Sitoutumisen ja motivaation lisääntyminen
Khan Academy	Pelillistämisen tuloksellisuus ei sisälly tutkimukseen
Order of the Curmudgeons	Lisääntynyt sitoutuminen ja opetusmateriaalin ymmärrys, parantuneet arvosanat

Taulukko 2. Tutkimusten raportoimat pelillistämiseen tulokset

4.1. Extreme programming -kurssi

Akpolat ja Slany [2014] jakavat pelillistetyn kurssin tulokset kahteen ryhmään: opiskelijoiden subjektiiviset tuntemukset ja tutkijoiden havainnoimat tulokset.

Akpolat ja Slany [2014] seurasivat opiskelijoiden toimintaa koodirivien lukumäärän, kommitointien lukumäärän avulla. Lisäksi opiskelijoiden piti ilmoittaa versionhallintajärjestelmään syötetyissä kommiteissa, mitä XP-käytäntöä kommitin tekemisessä oli seurattu. Havaintoina näistä tiedoista oli, että kun jokin XP-käytäntö oli viikon haasteena, opiskelijat hyödynsivät kyseistä käytäntöä enemmän. Toinen havainto oli, että kun jokin XP-käytäntö oli kerran ollut viikon haasteena, sen hyödyntäminen hiipui seuraavien viikkojen aikana, mutta ei kurssin loppuaikana tippunut viikkohaastetta edeltävälle tasolle. Jo kerran viikkohaasteessa olleen XP-käytännön ottaminen myöhemmin uudestaan viikkohaasteen aiheeksi nosti jälleen kyseisen käytännön käyttöastetta. [Akpolat and Slany 2014]

Opiskelijoille järjestettyjen kyselyjen perusteella 79 % opiskelijoista arvioi oppimistuloksensa olevan erittäin hyvä tai hyvä verrattuna vastaaviin kursseihin ilman pelillistämistä. Viikoittaiset haasteet kannustivat ystävällismieliseen kilpailuun ryhmien välillä ja näin kasvattivat sitoutumista kurssin opiskeluun. [Akpolat and Slany 2014]

4.2. Redmine

Buismanin ja van Eekelenin [2014] tavoitteena oli selvittää, miten pelillistäminen vaikuttaa Redminen käyttöön ja oppilaiden motivaatioon, sitoutumiseen ja osallistumiseen. Opiskelijoille kurssin jälkeen järjestetyn kyselyn tulosten perusteella Buisman ja van Eekelen [2014] havaitsivat, että pelillistämiselementit eivät vaikuttaneet merkitsevästi motivaatioon, sitoutumiseen ja osallistumiseen kontrolliryhmän ja pelillistämiselementtejä käyttäneiden ryhmien välillä.

Kerättyjen pisteiden ja suoritettujen toimintojen määrään pelillistämisellä oli sen sijaan huomattava merkitys. Ryhmä, joka näki omat pisteensä ja pystyi vertaamaan niitä toisten pisteisiin, suoritti enemmän toimintoja ja keräsi enemmän pisteitä kuin toiset ryhmät. Kontrolliryhmän ja ryhmän, joka näki vain omat pisteensä välillä, ei ollut suurta eroa aktiivisuuden ja pistemäärien suhteen.

Buisman ja van Eekelen [2014] pohtivat, miksi pelillistämiselementtien käyttö ei näy sitoutumisena ja motivaationa. Selityksenä tälle he tarjosivat sitä, että tutkimuksessa käytettiin vähän pelillistämiselementtejä, lähinnä pisteitä, ja ne vaikuttavat lähinnä ulkoiseen motivaatioon, jonka vaikutus on usein lyhytkestoista. Lisäksi kysely, jolla sitoutumista ja motivaatiota mitattiin, järjestettiin

vasta kurssin loputtua, jolloin pelillistämisen vaikutukset ovat voineet jo hiipua.

Valitettavasti Buisman ja van Eekelen [2014] joutuivat jättämään kurssiarvosanojen ja pelillistämisen vaikutuksen vertailun pois tutkimuksesta, sillä he eivät voineet taata tilastoidun tiedon oikeellisuutta.

4.3. Kunniamerkit ja A+

Haaranen ja muut [2014] eivät havainneet pelillistämisellä juuri minkäänlaista vaikutusta kurssin tuloksiin tai opiskelijoiden käytökseen. Kurssin harjoitukset toimivat niin, että läpäisyyn vaadittiin tietty minimisuoritusmäärä tehtävistä ja halutessaan paremman arvosanan, tehtäviä piti tehdä enemmän. Jotkut opiskelijat lopettivat tehtävien tekemisen heti, kun olivat saavuttaneet haluamaansa arvosanaan oikeuttavan määrän suorituksia, jolloin kurssin loppua kohti tehtävien suorittajien määrä väheni. Kunniamerkit tulivat näkyviin vasta viidennestä harjoituksesta lähtien.

Opiskelijoille järjestetyn kyselyn tuloksien mukaan suhtautuminen kunniamerkkeihin vaihteli hyvin negatiivisesta hyvin positiiviseen, mutta enimmäkseen opiskelijat olivat välinpitämättömiä.

4.4. Q-Learning-G

Opiskelijoiden pääasiallinen tavoite oli kerätä 100 suorituspistettä, jotka tarvittiin kurssin läpäisyyn. Tutkimuksen tuloksista selviää, että suurin osa opiskelijoista jatkoi työskentelyä senkin jälkeen, kun nuo 100 suorituspistettä olivat kasassa. Yleisimpiä vastauksia siihen, miksi he jatkoivat, olivat se, että opiskelijat halusivat kerätä kaikki kunniamerkit ja toisaalta he tahtoivat jatkaa C-kielen opiskelua. Lähempi tarkastelu paljastaa, että 100 suorituspisteen saavuttamisen jälkeen ympäristössä tehtyjen toimintojen painopiste muuttui enemmän hupikäyttöön, kuten puuttuvien kunniamerkkien keräilyyn. Pelillistäminen näyttäisi siis lisänneen käyttäjien sitoutumista ja motivaatiota. [Ibáñez *et al.* 2014]

Tärkeänä tavoitteena oli myös opiskelijoiden C-kielen taitojen kartuttaminen. Ibáñez ja muut [2014] analysoivat ennen kurssia ja kurssin jälkeen järjestetyn C-kielitestin tulokset ja tulokset olivat tilastollisesti merkitsevästi paremmat jälkimmäisessä testissä. Valitettavasti he eivät tarjoa mitään verrokkia, johon tuloksia voisi suhteuttaa. Esitetyistä tuloksista voi päätellä vain, että opiskelijat oppivat C-kieltä myös pelillistetyssä ympäristössä.

4.5. Javala

Lehtonen ja muut [2014] havaitsivat, että käyttäjät, joille pelillistämiselementit näkyivät, käyttivät keskimäärin enemmän aikaa ja ratkaisivat keskimäärin enemmän tehtäviä verrattuna käyttäjiin, joille pelillistäminen ei näkynyt. Ilman

pelillistämiselementtejä aikaa yhteen istuntoon käytettiin keskimäärin 26 minuuttia ja tehtäviä ratkaistiin keskimäärin kolme. Pelillistämiselementtien kanssa aika nousi 41 minuuttiin ja tehtäviä ratkaistiin viisi. Lisäksi kun pelillistämiselementit olivat näkyvissä, 2 % käyttäjistä ratkaisi kaikki tehtävät; ilman pelillistämistä yksikään käyttäjä ei tehnyt kaikkia tehtäviä.

Ajankäytöllisesti niiden käyttäjien määrä, jotka jatkoivat istuntoaan pidempään kuin tunnin niin, että pitivät vain lyhyitä alle viiden minuutin taukoja, oli kaksi kertaa suurempi kuin ilman pelillistämistä. Lehtonen ja muut [2014] pitävät tätä merkkinä siitä, että käyttäjä on flow-tilassa.

4.6. Khan Academy

Morrison ja DiSalvo [2014] tarkastelivat, millaista pelillistämistä Khan Academy on toteuttanut ja miten sen pelillistämistä voisi parantaa. Heidän tulkintansa mukaan Khan Academy vetää puoleensa valmiiksi motivoituneita käyttäjiä, joiden sitoutumista ja eteenpäin pyrkimistä pyritään parantamaan toteutetuilla pelillistämismenetelmillä.

4.7. "Order of the Curmudgeons"

O'Donovan ja muut [2013] mittasivat pelillistämisen onnistumista opiskelijoille lähetetyllä kyselylomakkeella ja kurssin arvosanoja edellisiin vuosiin vertaamalla. Kyselylomakkeen tulokset olivat kaikilta osin positiivisia, mutta eniten pelillistäminen lisäsi opiskelijoiden mukaan ymmärrystään kurssimateriaalista ja erityisesti lisäsi heidän sitoutumistaan kurssiin. Käytetyistä pelillistämiselementeistä motivoivimpana opiskelijat pitivät pistetilastoa, tarinallisuus sai heikoimman arvion.

Sitoutumisen ja ymmärryksen lisääntymisen lisäksi kurssin arvosanat paranivat merkittävästi verrattuna edellisvuoteen, mutta O'Donovan ja muut [2013] eivät kuitenkaan voi pitää tätä suoraan pelillistämisen ansiona, sillä kurssien sisältö ja kurssitarjonta muuttuivat näiden vuosien välillä suuresti muiltakin osin.

Kolmanneksi luennoille osallistumisesta palkittiin ja se näkyi merkittävissä määrin kävijämäärissä. Kyseisen yliopiston tietotekniikkaa käsittelevien kurssien osallistujamäärä on keskimäärin 30-60 % kurssin kävijöistä, kun tämän kurssin osallistujamäärän keskiarvo oli 79 %. [O'Donovan *et al.* 2013]

Opiskelijoiden antamat pisteet kurssin luennoitsijoille olivat myös paremmat kuin edellisinä vuosina, mutta eivät tilastollisesti merkitsevästi.

4.8. Tulosten vertailu vastaaviin kirjallisuustutkimuksiin

Vihavainen ja muut [2014] käyvät läpi 60 tutkimusta, joissa on testattu ohjelmoinnin perusteiden opetusta vaihtoehtoisin tavoin, perinteisen luennot ja har-

joitukset -menetelmän sijaan. Tutkimuksista kerättiin kolmetoista erilaista lähestymistapaa, joiden vaikutus ohjelmoinnin perusteiden kurssin läpäisyyn arvioitiin. Pelillistäminen oli yksi näistä lähestymistavoista, eikä se pärjää kovin hyvin muihin lähestymistapoihin verrattuna, sillä se on suhteellisen tulosparannuksen suhteen vasta kymmenes. [Vihavainen *et al.* 2014]

Vihavaisen ja muiden [2014] tekstistä ei kuitenkaan pysty täysin tulkitsemaan, mitä he lukevat pelillistämiseen kuuluvaksi. Vaikka tutkimuksessa käytetään termiä pelillistäminen, se on tutkimuksessa kategorisoitu samaistuttavan sisällön ja kontekstualisoinnin (relatable content and contextualization) alle. He saattavat siis tarkoittaa pelkästään pelimateriaalin ja peliesimerkkien käyttämistä opetuksessa. Tämä epäselvyys hankaloittaa tulosten yhteensopivuutta tämän tutkielman kontekstiin. Vihavaisen ja muiden [2014] tutkimus kuitenkin käsittelee suoraan tämän tutkielman aihepiiriä ja vahvistaa, että pelillistämisellä on positiivinen vaikutus oppimistuloksiin isommallakin otannalla.

Hamari ja muut [2014] keräävät yhteen 24 tutkimusta ja etsivät niistä vastausta kysymykseen ”toimiiko pelillistäminen?”. Hamarin ja muiden [2014] mukaan suurin osa käsitellyistä tutkimuksista tuotti positiivisia vaikutuksia tai tuloksia. Tutkimuksiin liittyi Hamarin ja muiden [2014] mukaan myös joukko epäkohtia: 1) joissain tutkimuksissa osallistujamäärät olivat pieniä, 2) soveltuvia psykometrisiä mittaustekniikoita ei käytetty, 3) joistain tutkimuksista puuttui kontrolliryhmä ja pohjautuivat pelkästään käyttäjien arviointeihin, 4) pelillistämistekniikoiden vaikutuksia ei pyritty eristämään toisistaan, vaan niitä testattiin yhteisvaikutuksena, 5) tutkimusten aikavälit olivat monessa kohtaa hyvin lyhyitä, jolloin positiivinen vaikutus voi olla pelkästään uutuuden viehätyksen aiheuttamaa, 6) joissain tutkimuksissa tulosten raportointi ei ollut kovin selvää ja 7) yhdessäkään tutkimuksessa ei käytetty monitahoisia mittausmalleja, eikä kaikkia pelillistämisen menetelmiä.

Tämän tutkielman käsittelemistä tapauksista neljässä havaittiin lisääntynyttä sitoutumista tai motivaatiota sovelluksen käyttämiseen. Yhdessä tutkimuksessa havaittiin tutkitun pelillistetyn sovelluksen lisääntynyttä käyttöä, mutta ei havaittu sitoutumisen tai motivaation lisääntymistä. Kahdessa käsitellyistä tutkimuksista ei havaittu pelillistämisellä merkittävää vaikutusta tai vaikutuksia ei edes käsitelty.

Positiivisia tuloksia raportoineet tutkimukset extreme programming -kurssi, Redmine, Q-Learning-G, Javala ja Order of the Curmudgeons kaikki sisälsivät Hamarin ja muiden [2014] listaamia epäkohtia. Näitä epäkohtia käsiteltiin myös useimmissa tutkimuksissa avoimesti.

Extreme programming -kurssin tutkimus oli lyhyt, sillä se kesti kuusi viikkoa kurssin 14 viikon kestosta. Tämän lisäksi tutkimuksella ei ollut kontrolli-

ryhmää ja oppimistulosten arviointi perustui käyttäjien palautteeseen eikä esimerkiksi kurssin arvosanoihin. [Akpolat and Slany 2014]

Redminen tutkimus suoritettiin vain yhdelle kurssille ja tuloksien arviointiin käytetty kysely suoritettiin vasta kurssin loputtua, jolloin mahdollisesti lyhytkestoiset motivaatiovaikutukset olivat jo kadonneet. Tätä on kuitenkin mahdotonta todeta, koska kyselyitä ei suoritettu kurssin aikana. [Buisman and van Eekelen 2014]

Q-Learning-G-tutkimuksessa osallistujamäärä oli pieni, 22 opiskelijaa, eikä käytössä ollut kontrolliryhmää. [Ibáñez *et al.* 2014]

Javala-tutkimus ei pyri tutkimaan eri pelillistämismenetelmien vaikutuksia, joen ei ole selvä mikä tai mitkä osat pelillistetyssä kokonaisuudessa selittävät saavutetut tulokset. [Lehtonen *et al.* 2014]

Order of the Curmudgeons -tutkimuksen suurin puute on kontrolliryhmän puute, sillä tutkimus sijoittuu ajanhetkeen, jossa kurssisisältö muuttui muutenkin, jolloin aiempien kurssisuoritusten käyttäminen kontrollina ei onnistunut. [O'Donovan *et al.* 2013]

Yhtään tämän tutkielman tutkimuksista ei ole käsitelty yllä mainituissa Hamarin ja muiden [2014] tai Vihavaisen ja muiden [2014] kokoomatutkimuksissa.

5. Yhteenveto

Tässä tutkielmassa tarkasteltiin pelillistämisen käyttämistä ohjelmoinnin opetuksessa ja opiskelussa. Kirjallisuudesta valikoitiin seitsemän tutkimusta, joiden tuloksia käytiin läpi tarkemmin.

Tutkielman alussa käsiteltiin pelillistämisen määritelmää ja sen kanssa kilpailevia termejä ja sitä kohtaan esitettyä kritiikkiä. Pohjustukseksi esiteltiin myös pelillistämisen tekniikoita, joita kirjallisuudessa on käsitelty ja erityisesti huomioitiin opetuskäyttöön suunnattujen sovellusten käyttämät tekniikat.

Tutkiemassa läpikäydyt esimerkkitapaukset käsittelivät aihetta extreme programming -kurssi, Redmine, A+, Q-Learning-G, Javala, Khan Academy ja Order of the Curmudgeons. Näistä tutkimuksista tarkasteltiin tutkimuksen sisältöä ja selvitettiin, millaisia pelillistämisen tekniikoita ne käyttivät toteutuksissaan. Esimerkkitapausten tuloksia käsiteltiin ja havaittiin, että pelillistämisellä oli myönteisiä vaikutuksia, mutta tutkimusten menetelmissä havaittiin puutteita ja epäkohtia, joita myös laajemmat tutkimukset ovat havainneet.

Ohjelmoinnin opetuksen ja oppilasmäärien lisääntyessä tehokkaat opetusmenetelmät ovat tärkeä tutkimuskohde. Pelillistäminen ohjelmoinnin opetuksessa vaatii ehdottomasti lisää tutkimusta. Olisi esimerkiksi hyvä suorittaa sama tutkimus uudemman kerran useammalla kurssilla ja vaikka kontrolliryh-

män kera, niin että tulokset eivät jää vain yhden kurssin yhden suorituskerran tuottamaan dataan. Pidemmällä aikavälillä pystyisi myös selvittämään, ovatko saavutetut positiiviset vaikutukset perustuneet pelkkään uutuuden viehätykseen.

Viiteluettelo

Bilal Akpolat and Wolfgang Slany. 2014. Enhancing software engineering student team engagement in a high-intensity extreme programming course using gamification. In: *Proc. of the 27th Conference on Software Engineering Education and Training*, 149-153.

Achilleas Buisman and Marko van Eekelen. 2014. Gamification in Educational Software Development. In: *Proc. of the Computer Science Education Research Conference*, 9-20.

Iain Bogost. 2011. Persuasive Games: Exploitationware. http://www.gamasutra.com/view/feature/134735/persuasive_games_exploitationware.php . Accessed 25.5.2016

Sebastian Deterding. 2015. The Lens of Intrinsic Skill Atoms: A Method for Gameful Design. *Human-Computer Interaction* 30, 3-4, 294-335.

Sebastian Deterding, Dan Dixon, Rilla Khaled and Lennart Nacke. 2011. From game design elements to gamefulness: Defining "gamification". In: *Proc. of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, 9-15.

Fiona Fui-Hoon Nah, Qing Zeng, Venkata Rajasekhar Telaprolu, Abhishek Ayyappa and Brenda Eschenbrenner. 2014. Gamification of Education: A Review of Literature. In: *HCI in Business - Lecture Notes in Computer Science*, 8527, 401-409

Lassi Haaranen, Petri Ihantola, Lasse Hakulinen and Ari Korhonen. 2014. How (not) to Introduce Badges to Online Exercises. In: *Proc. of the 45th ACM Technical Symposium on Computer Science Education*, 33-38.

Juho Hamari, Jonna Koivisto and Harri Sarsa. 2014. Does Gamification Work? -- A Literature Review of Empirical Studies on Gamification. In: *Proc. of the 47th Hawaii International Conference on System Sciences*, 3025-3034.

María-Blanca Ibáñez, Ángela Di-Serio and Carlos Delgado-Kloos. 2014. Gamification for Engaging Computer Science Students in Learning Activities: A Case Study Learning Technologies. *IEEE Transactions on Learning Technologies 7*, 3, 291 – 301.

Michael Lee, Andrew Ko and Irwin Kwan. 2013. In-game assessments increase novice programmers' engagement and level completion speed. In: *Proc. of*

the Ninth Annual International ACM Conference on International Computing Education Research, 153-160.

Timo Lehtonen, Timo Aho, Essi Isohanni and Tommi Mikkonen. 2014. On the role of gamification and localization in an open online learning environment: javala experiences. In: *Proc. of the 15th Koli Calling Conference on Computing Education Research*, 50-59.

Jane McGonical. 2011. We don't need no stinkin' badges: How to re-invent reality without gamification. http://www.gdcvault.com/play/1014576/We-Don-t-Need-No . Accessed 25.5.2016

Briana Morrison and Betsy DiSalvo. 2014. Khan academy gamifies computer science. In: *Proc. of the 45th ACM Technical Symposium on Computer Science Education*, 39-44.

Juhani Mykkänen and Linda Liukas. 2014. Koodi2016 – Ensiapua ohjelmoinnin opettamiseen peruskoulussa. http://koodi2016.fi/ . Viitattu 25.5.2016

Siobhan O'Donovan, James Gain and Patrick Marais. 2013. A case study in the gamification of a university-level games development course. In: *Proc. of the South African Institute for Computer Scientists and Information Technologists Conference*, 242-251.

Arto Vihavainen, Jonne Airaksinen and Christopher Watson. 2014. A systematic review of approaches for teaching introductory programming and their influence on success. In: *Proc. of the Tenth Annual Conference on International Computing Education Research*, 19-26.

Christopher Watson and Frederick Li. 2014. Failure rates in introductory programming revisited. In: *Proc. of the 2014 Conference on Innovation & Technology in Computer Science Education*, 39-44.

Gabe Zichermann and Christopher Cunningham. 2011. *Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps*. O'Reilly.

Käyttäjäpalaute ohjelmistokehityksessä

Sanni Kurvi

Tiivistelmä

Käyttäjiltä saatavan palautteen avulla voidaan korjata sovelluksen mahdollisia virheitä sekä kehittää uusia ominaisuuksia ja näin säilyttää käyttäjien mielenkiinto. On siis tärkeää motivoida käyttäjiä antamaan laadukasta palautetta. Käyttäjän motivaatioon antaa palautetta vaikuttaa moni tekijä. Palautteenantometodin tulee esimerkiksi olla mahdollisimman yksinkertainen ja nopea, eikä palautepyyntöjä tule esittää liian usein. Palautetta voidaan kerätä aktiivisesti tai passiivisesti. Tyypillisiä palautteenkeruumetodeja ovat sähköposti, erilaiset kyselyt ja haastattelut sekä käytettävyystestaukset. Uudempia kanavia palautteen keräämiseen tarjoavat mobiililaitteiden sovelluskaupat sekä sosiaalinen media. Uusien kanavien myötä palautteiden määrät ovat kasvaneet, mikä aiheuttaa lisätyötä niiden analysoimiseen. Palautteiden analysoiminen manuaalisesti ei enää ole tehokas vaihtoehto, vaan tulisi selvittää automatisoituja vaihtoehtoja.

Avainsanat ja -sanonnat: käyttäjäpalaute, käyttäjän osallistaminen, käyttäjäkokemus

1. Johdanto

ISO 9241-210 -standardin [2010] mukaan käyttäjäkokemus kuvaa niitä henkilön havaintoja ja reaktioita, jotka syntyvät tuotteen, järjestelmän tai palvelun käytöstä tai odotetusta käytöstä. Käyttäjäpalaute liittyy vahvasti käyttäjäkokemukseen, sillä palaute annetaan nimenomaan kokemuksen perusteella. Sampsonin [1998] sekä Almalikin ja muiden [2014] mukaan sekä erityisen positiivinen että erityisen negatiivinen käyttäjäkokemus saavat käyttäjän antamaan palautetta.

Vermeeren ja muut [2010] huomauttavat, että käyttäjäkokemusta ei pitäisi arvioida vasta tuotteen käytön jälkeen, vaan tulisi selvittää, mitä käyttäjät odottavat tuotteelta ennen käyttöä ja toteutuvatko odotukset käytön aikana. Lyhyen aikavälin tietojen lisäksi olisi tärkeää selvittää, miten käyttäjäkokemus kehittyy ajan myötä [Vermeeren *et al.* 2010]. Käyttäjäpalautetta kerätään usein siinä vaiheessa, kun tuote on julkistettu ja käyttäjät ovat päässeet sitä käyttämään. Kuitenkin myös ennen julkaisua tapahtuvasta käytettävyystestauksesta saatu palaute voidaan ajatella käyttäjäpalautteeksi, sillä onhan se saatu käyttäjiltä.

Käyttäjäkokemus on melko laaja käsite ja näin ollen myös käyttäjäpalautetta voi tulla hyvinkin erilaisista aiheista. Palaute voi olla esimerkiksi virheraportteja, palautetta sovelluksen ominaisuuksista tai toiveita uusista toiminnollisuuk-

sista [Pagano and Brügge 2013]. Tässä tutkielmassa keskitytään käsittelemään erilaisten sovellusten ja ohjelmistojen käyttäjäpalautetta ja niihin viitataan tutkielmassa sanoilla sovellus, ohjelma, ohjelmisto, tuote tai palvelu. Käyttäjällä tutkielmassa tarkoitetaan ohjelmiston loppukäyttäjää eli niitä käyttäjiä, jotka tosiasiallisesti tulevat ohjelmistoa käyttämään. Käyttäjiin viitataan sanoilla käyttäjä tai asiakas.

On tärkeää selvittää, mitkä menetelmät ovat käyttäjille mieluisimpia, jotta saadaan mahdollisimman paljon mahdollisimman hyödyllistä palautetta. Toisaalta on myös selvitettävä, millä tavoilla saadaan kerättyä kehittäjille erityisen hyödyllistä ja helposti analysoitavaa tietoa. Saatua palautetta käytetään tuotteen tai palvelun virheiden korjaamiseen sekä jatkokehitykseen. Näin ollen, mikäli palautetta ei saada, se on laadultaan huonoa tai sitä analysoidaan väärin, voi tuotteen jatkokehitys epäonnistua.

Tutkielman toisessa luvussa käydään läpi käyttäjäpalautteen merkityksiä ja tarkoitusta. Kolmannessa luvussa käsitellään tekijöitä, jotka vaikuttavat käyttäjien motivaatioon antaa palautetta. Neljännessä luvussa puhutaan erilaisista tavoista kerätä palautetta ja viidennessä luvussa käsitellään saadun palautteen analysointia ja hyödyntämistä. Kuudennessa luvussa käsitellään palautteen analysoinnin automatisointia. Tutkielman tarkoitus on selvittää lukijalle käyttäjäpalautteen tärkeyttä ja osoittaa, että sen keräämiseen ja analysointiin on kiinnitettävä erityistä tarkkuutta, jotta palautteesta on aidosti hyötyä.

2. Käyttäjäpalautteen merkitys

Mitä käyttäjäpalautteella halutaan saavuttaa eli toisin sanoen, miksi sitä kerätään ja miksi se on tärkeää? Paganon ja Brüggen [2013] mukaan palaute voi auttaa kehittäjiä parantamaan ohjelmiston laatua, tunnistamaan puuttuvia ominaisuuksia sekä selvittämään, miten ohjelmistoa oikeasti käytetään. Palaute auttaa myös arvioimaan ohjelmiston ja sen ominaisuuksien hyväksyntää käyttäjäyhteisön parissa. Myös Sherief [2014] kertoo, että käyttäjäpalautteen avulla voidaan tunnistaa ohjelmiston ongelmia, muokata nykyisiä vaatimuksia ohjelmistolle tai löytää lisävaatimuksia, jotka lisäävät käyttäjien tyytyväisyyttä.

Kaikkia käytettävyysongelmia ja virheitä ei aina löydetä ennen ohjelman tai palvelun julkaisua, joten kehittäjien on luotettava käyttäjäpalautteeseen saadakseen ne korjatuiksi [Heller *et al.* 2011]. Ohjelmiston arviointia on jatkettava julkaisun jälkeen myös siksi, että tietyn testiryhmän sijaan saataisiin palautetta useilta erilaisilta käyttäjiltä erilaisissa tilanteissa [Sherief 2014].

Yksinkertainen selitys käyttäjäpalautteen tarpeelle on se, että halutaan selvittää, miten tuotetta voidaan kehittää jatkossa entistä paremmaksi, jotta ihmiset edelleen jatkavat sen käyttöä. Ohjelmiston tulee täyttää käyttäjien tarpeet,

jotta siitä tulee yleisesti hyväksytty ja laajalti käytetty [Gärtner and Schneider 2012]. Pagano ja Brügge [2013] huomauttavat lisäksi, että käyttäjät ostavat tuotteen, joten luonnollisesti yritykset haluavat miellyttää heitä. Uusien käyttäjien hankkiminen onkin huomattavasti kalliimpaa kuin vanhojen käyttäjien pitäminen edelleen käyttäjinä [Bragge *et al.* 2005]. Uudelle käyttäjälle palvelua tai tuotetta pitää markkinoida, vanhoille se on jo entuudestaan tuttu eikä markkinointiin tarvitse käyttää resursseja. Käyttäjät voivat kuitenkin jopa vahingoittaa yritystä, jos he turhautuvat käyttämäänsä sovellukseen eivätkä saa yritykseltä vastakaikua [Pagano and Brügge 2013]. Näin ollen olemassa olevien käyttäjien palautetta on kannattavaa kuunnella.

Käyttäjäpalautteen avulla tehdyt parannukset voivatkin lisätä käyttäjien tyytyväisyyttä, kunhan osataan asettua käyttäjän asemaan ja tarkastellaan tuotetta sieltä käsin. Käyttäjä voi haluta jonkin toiminnon, mutta hän ei välttämättä halua erillistä laitetta toiminnon tuottamiseen. On myös erityisen tärkeää ymmärtää ja tietää, miten nykyinen tuote toimii ja miten tyytyväisiä käyttäjät siihen ovat ennen kuin suunnitellaan uusia tuotteita. [Fundin and Bergman 2003]

Esimerkiksi Braggen ja muiden [2005] mukaan yritykset tarvitsevat käyttäjiltä innovatiivista palautetta saadakseen uusia kehitysideoita. Käyttäjäpalaute voi auttaa myös tuotteen markkinoinnissa ilmaisemalla luottamusta positiivisen palautteen ja käyttäjäkokemusten muodossa [Pagano and Brügge 2013]. Näin voi tapahtua silloin, jos muiden käyttäjien antama palaute on julkisesti nähtävillä. Lisäksi palautteen avulla voidaan luoda keskustelua yrityksen ja asiakkaan välille [Sampson 1998].

Käyttäjät voivat palautteen avulla myös painostaa yritystä tekemään muutoksia. Palautteenantokanava valitaan tarkoituksellisesti eli mitä tärkeämpänä käyttäjät palautteenantamista pitävät, sitä julkisemman kanavan he valitsevat. Käyttäjät saattavat esimerkiksi antaa sovellukselle huonoja arvosanoja, jos yritys ei ole vastannut palautteeseen. [Pagano and Brügge 2013]

Seyff ja muut [2014] selvittivät, että esimerkiksi mobiilikehittäjät haluavat vastaanottaa ymmärrettävää ja helposti analysoitavaa palautetta. He tarvitsevat palautteen lisäksi myös tietoa muista käyttöön liittyvistä tekijöistä (akun taso, käyttöjärjestelmän versio, käyttäjän profiili jne.) ymmärtääkseen paremmin palautteen antamisen syytä. Ei siis riitä, että saadaan mahdollisimman paljon palautetta, vaan palautteesta on oltava konkreettista hyötyä.

Norman [2002] kertoo, että törmätessään ongelmaan ohjelman käytössä moni käyttäjä ei lähetä virheraporttia vaan ennemmin soimaa itseään ja opettelee jonkin tavan kiertää ongelma. Näin ollen käyttäjäpalautteen antamisen tulisikin olla mahdollisimman helppoa ja nopeaa ja käyttäjiä tulisi kannustaa sen antamiseen, jotta ongelmat tulisivat ilmi.

3. Käyttäjän motivaatio palautteen antamiseen

Olennainen tekijä käyttäjäpalautteen keräämisessä on käyttäjien motivaatio antaa palautetta. Käyttäjät haluavat antaa palautetta, kunhan se ei aiheuta heille suurta vaivaa [Pagano and Maalej 2013]. Motivaation puute voi johtaa siihen, ettei palautetta anneta lainkaan, mutta myös hyödyttömän palautteen antamiseen. Tällöin aikaa menee hukkaan sekä käyttäjältä että yritykseltä, joka joutuu tämän turhan palautteen käymään läpi. Näin ollen on tärkeää tietää, mikä käyttäjiä motivoi antamaan palautetta ja myös se, mitkä tekijät laskevat palautteen antamisen motivaatiota.

Käyttäjää voidaan kannustaa antamaan palautetta esimerkiksi konkreettisten palkkioiden ja houkutteiden, kuten alennuskuponkien tai arvontojen avulla. Tällaiset keinot, joilla hyvin selkeästi pyritään vaikuttamaan käyttäjään, ovat kuitenkin vain yksi palautteen antamisen motivaatioon vaikuttaja tekijä. Käyttäjän motivaatio antaa palautetta koostuu useista eri asioista, kuten käyttäjäkokemuksesta, palautteenantometodin käytettävyydestä, palautteen pyytämisen tavoista, muiden antamista palautteista ja siitä, saako palautteeseen vastauksen. Almaliki ja muut [2014] toteavat, että jotkin näistä palautteenannon motivaatioon vaikuttavista tekijöistä voivat vaikuttaa toisiinsa paljonkin. Esimerkiksi, jos käyttäjäkokemus oli erityisen hyvä, mutta palautetta pyydetään väärällä tavalla tai hetkellä, voi palaute jäädä antamatta.

3.1. Käyttäjäkokemus

Käyttäjän kokemus tuotteen tai palvelun käytöstä on avainasemassa palautetta annettaessa. Esimerkiksi Sampson [1998] esittää, että erityisen tyytyväiset tai erityisen tyytymättömät asiakkaat olisivat niitä, jotka antavat palautetta oma-aloitteisesti. Myös Almaliki ja muut [2014] selvittivät, että sekä hyvät kokemukset, mutta etenkin negatiiviset kokemukset voivat saada aikaan halun antaa palautetta. Tällöin näistä oma-aloitteisesti saaduista vastauksista olisi mahdollisesti helpompi saada nimenomaan hyödyllistä palautetta. Tyytymätön käyttäjä voi raportoida esimerkiksi erilaisista vioista tai antaa parannusehdotuksia, joista on konkreettista hyötyä tuotteen kehittäjille.

Käyttäjäkokemukseen vaikuttaa moni tekijä, kuten esimerkiksi käyttäjän odotukset tuotteen käytöstä [ISO 9241-210 2010; Vermeeren *et al.* 2010], joten siihen voi olla vaikea suoranaisesti vaikuttaa. Näin ollen kehittäjien tulisi keskittyä kokonaisuuteen ja selvittää käyttäjiltä, millaiseksi käyttäjäkokemukset lopulta todellisuudessa muotoutuvat. Palautteiden myötä voidaan tehdä muutoksia ja parannuksia, jotka taas osaltaan muokkaavat käyttäjäkokemuksia entistä paremmiksi.

3.2. Palautteenantometodin käytettävyys

Itse palautteen antamisen tulee olla mahdollisimman helppoa ja vaivatonta [Almaliki *et al.* 2014; Sampson 1998]. Mikäli palautteen antaminen on vaikeaa ja vaivalloista, asiakkaat todennäköisimmin jättävät vastaamatta [Sampson 1998]. Tyytymätön asiakas, joka ei onnistu antamaan palautetta, voi myös päätyä niin sanotusti äänestämään jaloillaan [Sampson 1998].

Palautteenkeräysmetodia suunniteltaessa on myös otettava huomioon, missä kontekstissa palautetta annetaan. Palautetta voidaan antaa sovelluksesta riippuen erilaisissa tilanteissa tai erilaisilla laitteilla. Esimerkiksi pitkän palautteen kirjoittaminen älypuhelimella on vaivalloisempaa kuin tietokoneella ja palaute voi sen takia jäädä antamatta [Almaliki *et al.* 2014]. Seyff ja muut [2014] selvittivätkin, että erityisesti älypuhelinten käytättäjille tärkeimpiä ominaisuuksia palautteenantometodille ovat helppokäyttöisyys sekä riittävä ohjeistus ja tuki.

Seyff ja muut [2014] kertovat myös, että käyttäjät haluaisivat erilaisia tapoja antaa palautetta; esimerkiksi tekstin lisäksi palautetta voisi olla mahdollista antaa myös audiona. Joissakin tilanteissa tai joillekin käyttäjäryhmille palautteen äänittäminen voi olla käyttäjälle parempi vaihtoehto, kuin tekstin kirjoittaminen. Seyffin ja muiden [2014] tutkimuksessa kävi kuitenkin ilmi, että tekstipalaute oli testikäyttäjille luonnollisempaa ja sitä annettiin enemmän kuin äänitteitä. Tuleekin harkita tapauskohtaisesti, onko tarpeen kerätä muuta, kuin tekstimuotoista palautetta.

3.3. Palautteen pyytäminen

Palautta pyydettäessä on huolehdittava siitä, että pyyntö on yksinkertainen ja vastaaminen on helppoa. Käytetyn kielen on oltava mahdollisimman selkeää ja ystävällistä. Lisäksi palautepyyntöön saatetaan vastata herkemmin, jos käyttäjille on perusteltu, miksi palautetta tarvitaan ja mitä sillä tehdään. [Almaliki *et al.* 2014]

Almaliki ja muut [2014] selvittivät, että erityisen tärkeää on palautteen pyytämisen ajoitus. He kertovat, että mikäli palautepyyntö tulee käyttäjälle väärällä hetkellä, voi tämä jättää kokonaan vastaamatta tai antaa huonolaatuista palautetta. Väärällä ajoituksella tarkoitetaan esimerkiksi sitä, että käyttäjä on tekemässä jotain muuta ja palautepyyntö keskeyttää tämän tekemisen. Käyttäjät eivät halua tulla keskeytetyksi, vaan palautteenantometodin tulisi olla mahdollisimman huomaamaton [Almaliki *et al.* 2014; Seyff *et al.* 2014]. Eräs Almalikin ja muiden [2014] ehdotus on välttää palautteen pyytämistä perinteisten toimistoaikojen aikana, sillä silloin käyttäjien kiireisyys voi vaikuttaa vastaamiseen. Tässä ongelmia voivat kuitenkin aiheuttaa eri aikavyöhykkeet ja kulttuurierot,

joista pitäisi ottaa selvää, mikäli palautepyynnöt halutaan ajastaa toimistoaikojen ulkopuolelle. Tulee myös selvittää, sopiiko kyseinen tapa tuotteen kohderyhmälle, eli ovatko he tyypillisimmin töissä juuri tavanomaisina toimistoaikoina. Myös esimerkiksi sähköpostitse palautteen pyytäminen antaa käyttäjille enemmän aikaa vastaamiseen eikä se yleensä keskeytä muita töitä [Almaliki *et al.* 2014] verrattuna vaikkapa mobiilisovellusten palautepyyntöihin, jotka peittävät osan puhelimen näytöstä. Tässä tulee kuitenkin huomata, ettei palautepyyntö saa olla liian huomaamaton, jotta palautetta kuitenkin saadaan. Palautepyynnön voisikin mahdollisuuksien mukaan ajoittaa heti sovelluksen käynnistämisen alkuun, jolloin se ei keskeytä jo aloitettua tekemistä. Almaliki ja muut [2014] huomauttavat myös, että palautteen pyytäminen tulisi ajoittaa siten, että käyttäjällä on ollut riittävästi aikaa tutustua tuotteeseen.

Koska käyttäjällä ei välttämättä ole aikaa vastata palautteeseen juuri silloin, kun sitä pyydetään [Almaliki *et al.* 2014], olisi hyvä kertoa käyttäjälle, miten kauan palautteen antamiseen suunnilleen kuluu aikaa sekä mahdollistaa palautteen antaminen myös myöhemmin. Tällöin käyttäjä voi palata vastaamaan silloin, kun se hänelle sopii. Seyffin ja muiden [2014] mukaan palautteen antamisen pitäisikin olla mahdollista käyttäjille juuri silloin, kun he haluavat, riippumatta ajasta ja paikasta.

On myös tärkeää, ettei palautetta pyydetä liian usein. Liian tiuhaan ajoitetut palautepyynnöt voivat johtaa siihen, että ne jätetään kokonaan huomiotta tai käyttäjä saattaa jopa ärsyyntymisen vuoksi lopettaa ohjelmiston käytön. Osa käyttäjistä voi kuitenkin tarvita muistutusta palautteenantamisesta. Käyttäjille pitäisikin olla mahdollisuus valita, millä tavalla ja miten usein he vastaanottavat pyyntöjä. [Almaliki *et al.* 2014]

3.4. Vuorovaikutuksellisuus

Muiden käyttäjien palautteiden näkeminen sekä se, että yritys vastaa käyttäjien palautteeseen, vaikuttavat käyttäjien motivaatioon antaa palautetta. Omasta mielipiteestä eriävän palautteen näkeminen voi lisätä käyttäjän motivaatiota antaa palautetta [Almaliki *et al.* 2014]. Myös sillä voi olla vaikutusta, onko palautetta jo annettu paljon. Esimerkiksi Almaliki ja muut [2014] kertovat, että jos palautetta on annettu vasta vähän, se voi lisätä halukkuutta antaa palautetta. Muiden palautteet voivat myös kertoa kannattaako kyseistä sovellusta ladata eli onko se käyttäjäyhteisön mielestä korkealaatuinen [Pagano and Maalej 2013].

Bajic ja Lyons [2011] kertovat, että sosiaalisessa mediassa käyttäjät todennäköisimmin jatkavat osallistumista, jos yritys arvostaa heidän kontribuutiotaan. Myös Almaliki ja muut [2014] selvittivät, että palautteen vaikutuksen näkeminen käytetyssä järjestelmässä lisää käyttäjien motivaatiota antaa palautetta.

Sampson [1998] mainitsee, että yksi käytetty vetoamismuoto palautepyynnöissä on lupaus siitä, että joku yrityksestä vastaa käyttäjän antamaan palautteeseen. Ei ole välttämättä merkityksetöntä sekään, kuka palautteeseen vastaa. Sampson [1998] kertoo, että paperilomakkeissa vastaanottajaksi on monesti merkitty jokin yrityksen johtohenkilö. Tällä tosin ei välttämättä ole merkitystä web-lomakkeissa, sillä niitä ei välttämättä osoiteta erityisesti kellekään tai ne käsitellään esimerkiksi asiakaspalvelussa. Myös Seyff ja muut [2014] sanovat, että käyttäjille olisi hyvä kertoa, kenelle palaute menee ja mitä sillä tehdään tai mitä sen avulla halutaan saavuttaa. Myöhemmin käyttäjille voidaan sovelluksen päivitysten yhteydessä kertoa, mitä muutoksia on tehty ja mikä oli palautteiden vaikutus näihin muutoksiin [Pagano and Maalej 2013].

Tärkeää on myös harkita, miten palautteisiin vastataan ja varmistaa, että vastaukset ovat riittävän personoituja. Automatisoitu vastausviesti varmistaa, että asiakas tietää palautteen tulleen perille, mutta voi syntyä tunne siitä, ettei viestiä kukaan kuitenkaan lue [Sampson 1998]. Web-lomakkeisiin voidaan sisällyttää kysymys siitä, haluaako palautteenantaja, että hänen palautteeseensa vastataan.

Palautteen antaminen voidaan siis nähdä myös sosiaalisena toimintana, mutta toisaalta osalle käyttäjistä on tärkeää pystyä vastaamaan anonyymisti [Almaliki *et al.* 2014; Sampson 1998]. Tällöin yritys ei kuitenkaan voi vastata palautteeseen.

4. Käyttäjäpalautteen keräämisen metodeista

Käyttäjäpalautetta voidaan kerätä joko aktiivisesti tai passiivisesti. Palautetta voidaan siis pyytää käyttäjiltä erikseen esimerkiksi sähköpostin, käytettävän ohjelman tai haastattelun kautta tai palautetta ei erikseen pyydetä, vaan esimerkiksi yrityksen tai tuotteen verkkosivuilla on kerrottu, miten palautetta voi antaa. Edellisessä luvussa käsiteltiin jonkin verran aktiivisen palautteen keräämisen ongelmia. Passiivisesti tehtävän käyttäjäpalautteen keräämisen etuna ovat sen alhaiset kustannukset, mutta toisaalta yritys ei voi juurikaan kontrolloida vastaajien määrää tai sitä, millaiset käyttäjät palautetta antavat [Sampson 1998].

Kuten jo aiemmin havaittiin, jättävät asiakkaat todennäköisesti vastaamatta, mikäli palautteen antaminen on vaikeaa [Sampson 1998]. Näin ollen palautteenkeruumetodin on ennen kaikkea oltava käyttäjälle mahdollisimman vaivaton. Braggen ja muiden [2005] mukaan palautteenkeruun tulisi olla suunniteltua, mutta käytetty metodi ei kuitenkaan saa rajoittaa vastaajia liikaa. Metodin tulisi myös kannustaa käyttäjiä uusien ideoiden keksimiseen [Bragge *et al.* 2005]. Käytetyn keräysmetodin tulisi olla toistettavissa ja muutettavissa siten,

että samaa tapaa voidaan käyttää uudelleen ja myös toisenlaisissa konteksteissa [Bragge *et al.* 2005]. Sen tulisi siis esimerkiksi toimia sekä tietokoneella että älypuhelimella. Pagano ja Brügge [2013] huomauttavat lisäksi, että kehittäjien tulisi voida vastata käyttäjien antamaan palautteeseen.

Uusia metodeja palautteen keräämiseen kehitetään, jotta sitä olisi helpompi ja nopeampi kerätä ja saataisiin kehittäjille mahdollisimman hyödyllistä palautetta. On myös tavallista, että palautetta saadaan useampien eri palautteenkeruukanavien kautta [Pagano and Brügge 2013]. Älypuhelinten yleistymisen myötä myös mobiilisovellukset ovat yleistyneet ja niiden palautteesta suuri osa kerätään sovelluskauppojen arviointijärjestelmän kautta. Myös sosiaalisen median rooli palautteen keräämisessä on entisestään kasvanut ja monet yritykset saavat suuren osan palautteestaan sitä kautta.

Seuraavaksi esitellään joitakin perinteisistä käyttäjäpalautteen keräämisen tavoista ja kerrotaan lisää sovelluskaupoista ja sosiaalisesta mediasta palautteenantokanavina. Tämän jälkeen esitellään vielä kaksi hieman uudempaa kokeilua käyttäjäpalautteen keräämiseksi.

4.1. Perinteisiä menetelmiä

Perinteisiä tapoja palautteen keräämiseen ovat esimerkiksi sähköpostiviestit, erilaiset kyselyt ja palautelomakkeet, haastattelut sekä ennen ohjelman julkaisua tehtävät käytettävyystestit. Bajic ja Lyons [2011] havaitsivat, että moni yritys pitää edelleen sähköpostitse saatuja palautteita muita, uudempia kanavia laadukkaampana. Tämä voi johtua esimerkiksi siitä, että sähköpostin kirjoittamiseen nähdään kenties enemmän vaivaa sitten, kun sitä viimein päätetään lähettää. Lisäksi edellisen luvun perusteella voidaan olettaa, että erityisen tyytyväinen tai tyytymätön käyttäjä lähettää palautetta pyytämättäkin.

Palautekyselyllä tarkoitetaan tässä erilaisia käyttäjille suunnattuja kyselyitä, gallupeja ja palautelomakkeita, joissa on ennalta määritellyt kysymykset, joihin käyttäjien halutaan vastaavan. Internetin kautta tehtävien kyselyiden etu on siinä, että ne on helppo ja nopea saada laajankin vastaajajoukon ulottuville. Ongelmana voidaan kuitenkin pitää sitä, että kyselyt voivat ohjata vastamaan vain tietystä, ennalta määritellystä näkökulmasta käsin [Bragge *et al.* 2005]. Ne siis keskittyvät keräämään palautetta ennalta määritellyistä aiheista, jotka ovat tärkeitä kyselyn laatijoiden mielestä, mutta eivät välttämättä jätä tilaa vastaajille [Bragge *et al.* 2005]. Aiemmassa luvussa todettiin palautepyyntöjen kohdalla, että käytetyllä kielellä on käyttäjille suuri merkitys. Tämän voidaan ajatella koskevan myös kyselyitä ja onkin huolehdittava, että vastaajat ymmärtävät kysymyksen oikein eikä kysymyksen asettelu johdattele vastaajaa liikaa.

Käyttävyystestien avulla saadaan usein runsaasti tietoa testattavan tuotteen käytettävyydestä ja siitä, miten käyttäjät palvelua käyttäisivät. Niiden ongel-

mana ovat kuitenkin korkeat kustannukset [Bragge *et al.* 2005]. Testien suunnittelu, pilotointi, testihenkilöiden valinta, testin toteutus ja tulosten analysointi vievät aikaa. Käytettävyystesteissä testikäyttäjien määrä on rajallinen, eivätkä käyttäjät toimi heille luonnollisessa ympäristössä käyttäessään testattavaa tuotetta [Hilbert and Redmiles 2001]. Sherief [2014] toteaakin, että perinteisissä käyttäjän osallistamisen tavoissa yksi ongelma on siinä, että ei voida täysin ennustaa ja simuloida tuotteen todellisia käyttökonteksteja ja käyttötapoja.

4.2. Sovelluskaupat

Sovelluskaupoissa, kuten AppStore, Google Play ja Windows Phone Store, käyttäjät voivat arvostella lataamansa sovelluksen käyttäen arvostelussa tähtiä yhdestä viiteen ja lisäksi arvioida sovellusta sanallisesti. Nämä arvioinnit ovat julkisia ja näkyvät kaikille käyttäjille sekä sovelluksen kehittäjille. Arviointien lisäksi käyttäjät kirjoittavat myös toiveita uusista ominaisuuksista ja muita parannusehdotuksia. Sovelluskaupat toimivat alustana käyttäjien vaatimusten ja kokemusten keräämiseen, vaihtamiseen ja hallinnoimiseen [Pagano and Maalej 2013].

Sovelluskauppojen kautta saadut palauteviestit ovat usein lyhyitä [Fu *et al.* 2013; Pagano and Maalej 2013], minkä Fu ja muut [2013] toteavat johtuvan siitä, että palaute kirjoitetaan usein mobiililaitteella. Pagano ja Maalej [2013] huomauttavat myös, että monet tällaisista lyhyistä palautteista ovat kehittäjille hyödyttömiä. Lyhyet viestit vahvistavat myös tulkintaa siitä, että palautteen antamiseen ei haluta käyttää kovinkaan paljoa aikaa. Palautetta tulee paljon, mutta se voi suurilta osin olla siis melko pinnallista [Seyff *et al.* 2014]. Palautteen pituus on Paganon ja Maalejin mukaan [2013] lisäksi kytköksissä tähtiarvioihin. Eräs heidän esittämänsä oletus on, että käyttäjät kirjoittaisivat vähemmän ollessaan tyytyväisempiä.

Sovelluskaupoissa oleva palaute koskee aina tiettyä sovellusta. Sovelluksista voi kuitenkin olla eri versioita, joten saman sovelluksen saama palaute voi vaihdella eri ajanjaksojen välillä paljonkin [Fu *et al.* 2013]. Lisäksi Pagano ja Maalej [2013] havaitsivat, että palautetta tulee eniten muutamina julkaisun jälkeisinä päivinä, minkä jälkeen tahti hiipuu.

Parhaimmat tähtiarviot saaneet sovellukset saavat sovelluskaupoissa paremmin näkyvyyttä, mikä taas nostaa niiden latausmääriä [Pagano and Brügge 2013]. Saadulla palautteella on siis heti konkreettinen merkitys sovelluksen menestykselle. Kehittäjille tärkeää palautetta ovat virheraportit ja käyttäjäkokemukset, joiden avulla he voivat korjata sovelluksen virheitä. Paganon ja Maalejin [2013] mukaan sovelluskauppojen palautteissa käyttäjät eivät kuitenkaan usein kerro käyttäjäkokemuksestaan tarkemmin silloin, kun se on ollut negatiivinen. Tästä voidaankin päätellä, että nykyisellään sovelluskauppojen palaute-

systeemi ei ole riittävän hyödyllinen kehittäjille, jotta he voisivat parantaa sovelluksia [Pagano and Maalej 2013].

Sovelluskaupat voivat kuitenkin olla väylänä käyttäjien ja kehittäjien välillä, sillä niiden kautta kuitenkin saadaan runsaasti kehuja ja kritiikkiä päivittäin. Nykyisestä järjestelmästä kuitenkin puuttuu kaksisuuntaisuus, eli kehittäjät eivät voi vastata käyttäjien palautteeseen ja kysyä täydentäviä kysymyksiä tai kertoa, että ongelma on korjattu. Tätä voidaan kiertää siten, että sovelluksen uusissa päivityksissä kerrottaisiin uudet muutokset ja niihin vaikuttaneet palautteet. Tämä lisäisi käyttäjien osallistamista ja auttaisi käyttäjiä ymmärtämään, miksi muutokset on tehty. [Pagano and Maalej 2013]

4.3. Sosiaalinen media

Sosiaalisen median käyttö on yleistynyt, ja yksityishenkilöiden lisäksi siellä ovat myös monet yritykset. Sosiaalinen media on edullinen ja samaan aikaan tehokas väylä palautteen keräämiseen ohjelman koko elinkaaren ajan [Bajic and Lyons 2011]. Sovelluskaupoista poiketen kehittäjillä on sosiaalisessa mediassa mahdollisuus vastata saatuun palautteeseen. Sosiaalisessa mediassa käyttäjät voivat jakaa ideoitaan ja parannusehdotuksiaan ja lisäksi yrityksen kommunikointi käyttäjien kanssa sosiaalisessa mediassa auttaa luomaan käyttäjille tunteen yhteisöllisyydestä [Bajic and Lyons 2011].

Voisi ajatella, että yksi sosiaalisen median etu perinteisempiin menetelmiin verrattuna on se, että käyttäjät ovat siellä jo valmiiksi, eikä heitä tarvitse erikseen houkutella sinne. Perinteisten menetelmien kohdalla käyttäjä myös tietää olevansa arvioinnin kohteena [Bajic and Lyons 2011], eikä välttämättä halua vastata kyselyihin [Hilbert and Redmiles 2001].

Bajic ja Lyons [2011] puhuvat sosiaalisista asiakaspalautteen hallintatyökaluista ja esittelevät tarkemmin UserVoice -palvelun. Palvelussa käyttäjät voivat jakaa ideoita sekä äänestää ja kommentoida muiden tekemiä ehdotuksia. Yritys voi vastata palautteisiin niin ikään kommentoimalla sekä lisäksi asettaa ehdotukselle statuksen, joka kertoo, onko ehdotus otettu tarkastelun alle, onko sitä alettu toteuttaa tai onko se jo toteutettu. Bajic ja Lyons [2011] huomasivat kuitenkin, että ehdotuksia ei aina toteutettu niin kuin käyttäjät olivat äänestäneet. He ehdottivat tälle syyksi sitä, että yritykset toteuttivat ensin mieluummin sellaisia ehdotuksia, joiden toteutus oli helpointa, vaikkei kyseinen ehdotus olisi ollut käyttäjien keskuudessa erityisen suosittu. Bajic ja Lyons [2011] kuitenkin huomauttivat, etteivät eniten ääniä ja kommentteja saaneet ehdotuksetkaan välttämättä ole aina tärkeimpiä, vaan niiden kannattajajoukko vain voi olla muita äänekkäämpi.

Erilaiset yritykset käyttävät sosiaalista mediaa eri tavoin ja eri tarkoituksiin. Pienet yritykset ja startup-yritykset käyttävät sosiaalista mediaa levittämään

sanaa, seuraamaan kilpailua sekä rakentamaan yhteisöä tuotteensa ympärille. Sen avulla yritetään myös selvittää, ollaanko tuotteen kehityksessä menossa oikeaan suuntaan. Suuret yritykset sen sijaan käyttävät sosiaalista mediaa teknisen tuen tarjoamiseen ja ennemminkin jo olemassa olevan suhteen huoltamiseen käyttäjäyhteisössä. Ne haluavat selvittää, mitä ihmiset niistä ajattelevat. [Bajic and Lyons 2011]

Sosiaalisen median huonona puolena voidaan pitää sovelluskauppojen tapaan palautteen suurta määrää. Suuren tietomäärän hallinta ja hyödyllisen tiedon poimiminen sen seasta voi olla hankalaa [Bajic and Lyons 2011]. Pagano ja Brügge [2013] huomauttavat myös, että käyttäjät voivat painostaa yritystä tekemään heidän haluamiaan muutoksia antamalla julkisesti huonoa palautetta.

4.4. Kokeellisia menetelmiä

Käyttäjäpalautteen keräämiseen kehitetään myös aivan uusia tapoja. Heller ja muut [2011] ovat esimerkiksi pilotoineet metodia, jossa käyttäjällä olisi palautteenantoa varten käytössään fyysinen painike. Tätä painiketta käyttäjä voi heti ongelman kohdatessaan ja turhautuessaan painaa ja hänelle avautuu tekstikenttä palautteen kirjoittamiseksi. Painike liitettäisiin esimerkiksi kannettavaan tietokoneeseen. Ehdotuksessa on omat ongelmansa alkaen painikkeen mukana kuljettamisesta, mutta Heller ja muut [2011] toteavatkin, että tavoite on löytää tasapaino siinä, että palautetta on mahdollisimman helppo antaa ja se olisi samalla kuitenkin mahdollisimman hyödyllistä kehittäjille.

Seyff ja muut [2014] ovat kehittäneet uutta metodia mobiilisovellusten palautteenantoon. He toteavat, että erilaisten mobiilisovellusten kysyntä on suurta ja myös käyttäjien vaatimukset sovelluksista ovat korkealla. Käyttäjien muuttuvien vaatimusten ja toiveiden vuoksi käyttäjäpalautteen kerääminen on erityisen tärkeää [Seyff *et al.* 2014]. He painottavat tarvetta sellaiselle palautteenantometodille, jonka kautta palautetta voitaisiin antaa heti ongelman ilmetessä. Perinteisten palautteenantometodien (esimerkiksi haastattelut) ongelmina Seyff ja muut [2014] pitävät pienen vastaajaryhmän lisäksi juuri sitä, ettei palautetta voi antaa heti ongelman tapahtuessa.

AppEcho on Seyffin ja muiden [2014] kehittämä palautemetodi älypuhelimille. Sovellus tallentaa tietoa käyttökontekstista ja lisäksi käyttäjän itse antaman palautteen. Sen avulla käyttäjiltä pyritään keräämään lyhyitä ja yksinkertaisia palautteita. AppEcho on kehitetty Samsung Bada OS ja Android OS -käyttöjärjestelmille. Tarkoitus on, että kohdatessaan ongelman tai löytäessään parannettavaa, käyttäjä ottaa tilanteesta kuvakaappauksen ja sen jälkeen avaa AppEcho-sovelluksen. Käyttäjä korostaa kuvakaappauksesta sen kohdan, jota palaute koskee ja kirjoittaa palautteen. Palaute lähetetään sähköpostitse ennalta määritetylle vastaanottajalle. Menetelmää testattuaan Seyff ja muut [2014] ha-

vaitsivat, että palautteen antamiseen vaikuttaa eri älypuhelimen käyttötavat eli esimerkiksi tehokäyttäjät antavat enemmän palautetta kuin peruskäyttäjät. He havaitsivat, että AppEchon tyyppinen palautemetodi tukee käyttäjiä antamaan palautetta ongelman havaitsemishetkellä. Huomattiin myös, että suuri osa palautteesta oli kehittäjille ymmärrettävää ja hyödyllistä, vaikkei heillä ollut mahdollisuutta kysyä tarkentavia kysymyksiä. Kahdensuuntaista kommunikointiväylää ei tarjottu käyttäjien yksityisyyden suojaamiseksi käyttäjien toiveiden mukaisesti.

5. Käyttäjäpalautteen analysointi ja hyödyntäminen

Käyttäjäpalaute sisältää tärkeää tietoa kehittäjille, sillä sen avulla voidaan parantaa sovelluksen laatua ja tunnistaa puuttuvia ominaisuuksia [Pagano and Brügge 2013]. Käyttäjäpalautteesta voi olla hyötyä myös kokonaan uusien tuotteiden kehittämisessä, vaikka käyttäjät eivät välttämättä osaa suoraan kuvailla niitä tuotteita, joita he haluavat tai tarvitsevat [Fundin and Bergman 2003]. Palautteen kerääminen ei siis yksinään riitä, vaan palaute on myös analysoitava mahdollisimman hyvin. Hyötyäkseen käyttäjäpalautteesta kehittäjien täytyy ymmärtää, mitä käyttäjä tarkoittaa, onko palaute relevanttia ja kuinka korkealle se tulisi priorisoida [Alkadhi *et al.* 2014; Pagano and Brügge 2013].

Käyttäjäpalautetta analysoidessa on tärkeää tiedostaa, miten palaute on kerätty, sillä käyttäjäkokemukset voivat vaihdella käyttökontekstista riippuen. Vermeeren ja muut [2010] sanovatkin, että olisi tärkeää kerätä tietoa käyttäjäkokemuksesta sellaisessa ympäristössä, joka vastaa mahdollisimman hyvin tuotteen todellisen käytön ympäristöä. Samalle ohjelmalle voidaan myös kerätä käyttäjäpalautetta useiden eri kanavien kautta [Pagano and Brügge 2013]. Eri kanavien kautta annettu palaute voi erota toisistaan esimerkiksi silloin, jos samaa sovellusta voidaan käyttää sekä tietokoneella, että mobiililaitteella. Tällöin ominaisuudet eroavat toisistaan ja palautekin on erilaista. Palautteissa on muitakin eroja ja erityyppiset palautteet hyödyttävät yritystä eri tavoin. Käyttäjien ehdotukset uusista ominaisuuksista voivat auttaa tuotteen jatkokehityksessä, kun taas virheraporttien avulla korjataan suoranaisia vikoja.

Palautteen analysointiin vaikuttaa myös saadun palautteen laatu. Paganon ja Brüggen [2013] mukaan monet käyttäjät antavat palautetta hyvinkin nopeasti ongelman ilmenemisen jälkeen suunnittelematta palautetta sen tarkemmin. Tämä voi johtaa epäselvään palautteeseen, mikä vaikeuttaa sen analysointia. Kehittäjät joutuvat lukemaan yksittäisiä palautteita useita kertoja ymmärtääkseen ne kunnolla [Pagano and Brügge 2013].

Pagano ja Brügge [2013] kertovat eniten vaivaa aiheutuvan siitä, että kehittäjien täytyy arvioida palautteen vaikutusta ja tärkeyttä. Tämä johtuu heidän

mukaansa siitä, että kehittäjien on manuaalisesti arvioitava, kuinka moneen käyttäjään tietty ominaisuus vaikuttaa ja kuinka moni todellisuudessa hyötyisi jostakin uudesta ominaisuudesta. Eri palautetyypit ovat myös arvoltaan eri asemassa, sillä esimerkiksi virheraportit ovat kriittisempiä kuin uudet ominaisuudet ja ne on näin ollen hoidettava nopeammin [Pagano and Brügge 2013]. Uusista ominaisuuksista päätettäessä taas on arvioita, onko ehdotus järkevä, parantaako se tuotetta ja sopiiko se tuotteen kehityskaareen [Pagano and Brügge 2013].

Yksi tärkeimmistä seikoista niin palautteen keräämisessä kuin analysoimisessa on saadun datan määrä. Vermeerenin ja muiden [2010] mukaan monesti ajatellaan, että mitä enemmän dataa sen parempi, mutta tämä ei välttämättä pidä paikkaansa. Suuremman tietomäärän kerääminen voi vaatia enemmän resursseja ja aikaa, eikä kehittäjillä lopulta välttämättä ole edes aikaa käydä kaikkea tätä dataa läpi siten, että sitä voitaisiin käyttää hyödyksi tuotteen kehityksessä [Guzman *et al.* 2014; Gärtner and Schneider 2012; Vermeeren *et al.* 2010]. Vermeeren ja muut [2010] myös huomattavat, että suuremman tietomäärän kerääminen voi usein vaatia enemmän vaivaa myös palautteenantajilta (esimerkiksi pitkät kyselyt tai haastattelut) ja tämä taas voi aiheuttaa heissä uupumusta, eikä kerätty data välttämättä enää olekaan luotettavaa.

Alkadhin ja muiden [2014] mukaan tagien eli asiasanojen käyttö voi auttaa kehittäjiä palautteen analysoimisessa. Tässä ehdotuksessa palautteet olisivat siis julkisesti kaikkien näkyvillä ja niihin voitaisiin liittää asianmukaisia asiasanoja. Asiasanat voitaisiin listata näkyville esimerkiksi sen mukaan, kuinka usein samaa asiasanaa on käytetty. Käyttäjien kanssa yhteistyössä käytettynä asiasanat voivat auttaa ryhmittelemään toisiinsa liittyviä palautteita yhteen, tekemään palautteista ymmärrettävämpiä ja huomaamaan nopeammin, mitkä ohjelmiston osat ja ominaisuudet saavat eniten palautetta [Alkadhi *et al.* 2014]. Alkadhi ja muut [2014] kuitenkin huomauttavat, että asiasanojen käytön positiiviset vaikutukset riippuvat siitä, miten hyvin ne on asetettu eli kuvaavatko ne palautetta oikein.

6. Automatisointi käyttäjäpalautteen analysoinnin avuksi

Manuaalisesti tehtynä analysointi vaatii paljon aikaa ja vaivaa. Olisikin kehitettävä automatisoituja keinoja analysoinnin helpottamiseksi [Gärtner and Schneider 2012; Pagano and Brügge 2013]. Pagano ja Brügge [2013] toteavat, että tarvitaan sellaisia työkaluja, jotka yhdistävät, jäsentävät, analysoivat ja jäljittävät käyttäjäpalautetta etenkin silloin, kun palautetta on paljon. Työkalun tulisi heidän mukaansa ryhmitellä samankaltaiset käyttäjäpalautteet, selvittää palautteen tyyppi (esimerkiksi virheraportti, parannusehdotus tai pyyntö uu-

desta ominaisuudesta) ja se, mihin ominaisuuteen palautteessa viitataan sekä antaa kehittäjille tietoa palautteen lähettäjästä (esimerkiksi onko käyttäjä ohjelman aktiivinen käyttäjä ja onko hän aiemmin antanut palautetta). Käyttäjäpalautetta systemaattisesti suodatettaessa ja analysoidessa on kuitenkin muistettava, että palautteet voivat sisältää myös sarkasmia: ”I lost all my phone contacts. Great, thank you!”, ja tämä on otettava huomioon suunniteltaessa automatisoitua analysointia [Pagano and Maalej 2013].

Palautteen analysointia voidaan helpottaa reaktiivisilla tai proaktiivisilla eli ennakoivilla työkaluilla. Reaktiivinen työkalu on sellainen, jota käytettäessä käyttäjäpalaute kerättäisiin kuten ennenkin ja työkalu analysoi saadun datan jälkikäteen. Proaktiivinen lähestymistapa voisi pyrkiä välttämään samanlaisen palautteen kertymistä tarjoamalla käyttäjälle mahdollisuuden äänestää jo annettua palautetta. Ennakoivat työkalut sopivat Paganon ja Brüggen [2013] mukaan käytettäviksi silloin, kun palautetta on paljon ja se on julkista. Reaktiivinen työkalu taas silloin, kun palautteen määrä on vähäisempi ja käyttäjät ovat ammattimaisempia loppukäyttäjiä. [Pagano and Brügge 2013]

Esimerkkejä kehitteillä olevista reaktiivisista työkaluista ovat esimerkiksi Gärtnerin ja Schneiderin [2012] kehittelemä ConTexter, joka tunnistaa palautteista tärkeitä aiheita ja priorisoi niitä automaattisesti. Toinen esimerkki on Fun ja muiden [2013] WisCom, jonka avulla voidaan analysoida sovelluskauppojen kautta saatavaa käyttäjäpalautetta. WisCom havaitsee epäjohdonmukaisuuksia tähtiarvioinnin ja kommenttien välillä, identifioi tärkeimpiä syitä siihen, miksei käyttäjät pidä sovelluksesta ja selvittää, miten palautteet muuttuvat ajan myötä. Kolmas esimerkki on Guzmanin ja muiden [2014] FAVe, joka on työkalu palautteen visualisointiin. FAVessa on erilaisia diagrammeja palautteista kerätyistä tiedoista. Näkymiä pystyy suodattamaan eri tavoin (aika, arvosana tuotteelle, ominaisuuden suosio) ja käyttäjä voi lisäksi tarkastella palautetta sen alkuperäisessä muodossa tai etsiä avainsanoja.

7. Yhteenveto

Käyttäjiltä saatu palaute on tärkeä osa ohjelmistokehitystä sovelluksen koko elinkaaren ajan. Palaute auttaa korjaamaan virheitä ja parantamaan tuotteen laatua.

Käyttäjän motivaatioon antaa palautetta vaikuttaa erityisesti käyttäjäkokemus, palautteenantometodin käytettävyys ja se, miten ja milloin palautetta pyydetään. Almalikin ja muiden [2014] mukaan palautetta ei pitäisi pyytää liian aikaisin, mutta ei myöskään liian myöhään tai usein. Voisikin olla syytä tutkia, mikä on otollisin aika palautteen pyytämiselle.

Palautteen keräämiseen on käytettävissä useita erilaisia metodeja ja uusia kehitetään edelleen. On tärkeää selvittää, mikä palautteenantometodi sopii parhaiten missäkin tilanteessa. Käyttäjät eivät ole valmiita käyttämään paljoa aikaa palautteen antamiseen, mutta he kuitenkin antavat palautetta. Pitäisikin panostaa siihen, että palautteen antamisen vaiva olisi minimoitu. Käyttäjiä tulee myös kannustaa jakamaan kokemuksiaan palvelusta [Pagano and Maalej 2013].

Palautteen analysoinnissa täytyy selvittää palautteen laatu, hyödyllisyys ja tärkeys. Manuaalisesti tämä vie paljon aikaa. Eniten aikaa vie palautteen ymmärtäminen. Palautteen analysointia voidaan kuitenkin helpottaa automatisoitujen työkalujen avulla. Käyttäjiä voisi myös opettaa antamaan hyödyllistä palautetta. Pagano ja Brügge [2013] kertovat, että käyttäjiä ei systemaattisesti opeteta antamaan tietynlaista yritykselle hyödyllistä palautetta. Virheraportteja lukuun ottamatta ei ole olemassa yleisesti hyväksyttyä tapaa sille, miten palautetta pitäisi antaa tai miten ohjelmiston evoluution aikana pitäisi kerätä palautetta.

Käyttäjäpalaute on siis tärkeää ja on kiinnitettävä huomiota siihen, miten sitä kerätään, jotta saataisiin mahdollisimman hyödyllistä palautetta. Palautteen antamisen motivaatioon vaikuttaviin tekijöihin tulee perehtyä, on valittu keräysmetodi mikä hyvänsä. On hyvä myös tiedostaa, että kaikki palaute ei ole arvokasta vaan laatu voi korvata määrän.

Tulevaisuudessa olisi mielenkiintoista tutkia, millä tavoin saadaan kaikkein hyödyllisintä palautetta. Voidaanko käyttäjiä opettaa antamaan entistä hyödyllisempää palautetta? Olisi myös mielenkiintoista selvittää, millaiset palautteenantometodit olisivat hyödyllisimpiä erilaisille erityisryhmille (esimerkiksi lapset, vanhukset, kehitysvammaiset), joille on jo kehitetty heidän tarpeisiinsa sopivia sovelluksia. Millaisia mahdollisuuksia antaa palautetta heille tällä hetkellä tarjotaan? Millaiset palautteenantometodit olisivat eri ryhmille toimivimpia? Kiinnostavaa on myös palautteen hyödyntäminen. Miten prosessi käytännössä toimii esimerkiksi erikokoisissa yrityksissä Suomessa? Lisäksi edelleen tulisi kehittää palautteen analysointia helpottavia tekniikoita, jotta käyttäjäpalautteen hyödyntäminen olisi tehokkaampaa.

Viiteluettelo

Rana Alkadhi, Dennis Pagano and Bernd Brügge. 2014. Can collaborative tagging improve user feedback? A case study. In: *Proc. of the 6th International Workshop on Social Software Engineering*, ACM, 1-8.

Malik Almaliki, Cornelius Ncube and Raian Ali. 2014. The design of adaptive acquisition of users feedback: an empirical study. In: *Proc. of the 2014 IEEE*

Eighth International Conference on Research Challenges in Information Science (RCIS), IEEE, 1-12.

Dejana Bajic and Kelly Lyons. 2011. Leveraging social media to gather user feed-back for software development. In: *Proc. of the 2nd International Workshop on Web 2.0 for Software Engineering*, ACM, 1-6.

Johanna Bragge, Hilkka Merisalo-Rantanen and Petri Hallikainen. 2005. Gathering innovative end-user feedback for continuous development of information systems: a repeatable and transferable e-collaboration process. *IEEE Transactions on Professional Communication* 48, 1, 55-67.

Bin Fu, Jialiu Lin, Lei Li, Christos Faloutsos, Jason Hong and Norman Sadeh. 2013. Why people hate your app: making sense of user feedback in a mobile app store. In: *Proc. of the 19th ACM SIGKDD International Conference on knowledge discovery and data mining*, ACM, 1276-1284.

Anders P. Fundin and Bo L.S. Bergman. 2003. Exploring the customer feedback process. *Measuring Business Excellence* 7, 2, 55-65.

Emitza Guzman, Padma Bhuvanagiri and Bernd Bruegge. 2014. FAVe: visualizing user feedback for software evolution. In: *Proc. of the 2014 Second IEEE Working Conference on Software Visualization,* IEEE, 167-171.

Stefan Gärtner and Kurt Schneider. 2012. A method for prioritizing end-user feedback for requirements engineering. In: *Proc. of the 5th International Workshop on Co-operative and Human Aspects of Software Engineering*, ACM, 47-49.

Florian Heller, Leonhard Lichtschlag, Moritz Wittenhagen, Thorsten Karrer and Jan Borchers. 2011. Me hates this: exploring different levels of user feedback for (usability) bug reporting. In: *Proc. of the CHI '11 Extended Abstracts on Human Factors in Computing Systems*, ACM, 1357-1362.

David M. Hilbert and David F. Redmiles. 2001. Large-scale collection of usage data to inform design. In: *Proc. of the 8th IFIP TC. 13 Conference on Human-Computer Interaction*, FXPAL.

ISO 9241-210:2010. *Ergonomics of human-system interaction – Part 210: Human-centered design for interactive systems.* International Standardization Organization (ISO).

Donald Norman. 2002. *The Design of Everyday Things*. Basic Books.

Dennis Pagano and Bernd Brügge. 2013. User involvement in software evolution practice: a case study. In: *Proc. of the 2013 International Conference on Software Engineering*, ACM, 953-962.

Dennis Pagano and Walid Maalej. 2013. User feedback in the appstore: an empirical study. In: *Proc. of the 2013 21st IEEE International Conference on Requirements Engineering (RE)*, IEEE, 125-134.

Scott E. Sampson. 1998. Gathering customer feedback via the Internet: instruments and prospects. *Industrial Management & Data Systems* 98, 2, 71-82.

Nada Sherief. 2014. Software evaluation via users' feedback at runtime. In: *Proc. of the 18th International Conference on Evaluation and Assessment in Software Engineering*, ACM, Article 58, 4 pages.

Norbert Seyff, Gregor Ollmann and Manfred Bortenschlager. 2014. AppEcho: a user-driven, in situ feedback approach for mobile platforms and applications. In: *Proc. of the 1st International Conference on Mobile Software Engineering and Systems*, ACM, 99-108.

Arnold P. O. S. Vermeeren, Effie Lai-Chong Law, Virpi Roto, Marianna Obrist, Jettie Hoonhout and Kaisa Väänänen-Vainio-Mattila. 2010. User experience evaluation methods: current state and development needs. In: *Proc. of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, ACM, 521-530.

Osaamisperustaisuutta korkeakoulutuksessa osaamisanalytiikan tukemana – teknisen toteutuksen ja toteutukseen liittyvien eettisten kysymysten tarkastelua

Toni Niittymäki

Tiivistelmä.

Tutkielmassa tarkastelen osaamisperustaisen arvioinnin tukemista oppimisanalytiikan avulla. Osaamisanalytiikaksi nimeämääni toimintaa lähestyn SCALA (Scalable Competence Assessment through a Learning Analytics Approach) -toteutuksesta julkaistujen tutkimusartikkeleiden avulla. Arvioin toteutusta teknisestä ja eettisestä näkökulmasta samalla sijoittaen tarkastelun niiden akateemisten tutkimusalojen kontekstiin, joita on muodostettu korkeakoulutuksessa tuotetun datan tutkimisen ympärille.

Tutkielman tuloksena pidän sitä, että osaamisanalytiikka oppimisanalytiikan osa-alueena edellyttää ja ansaitsee lisää huomiota ja jatkotutkimusta. Osaamisanalytiikkaan liittyvät tekniset ja eettiset kysymykset ovat ratkaistavissa. Suurimmat haasteet liittyvät pedagogisen prosessin rakentamiseen sellaiseksi, että eri osaamisten kehittymisestä tuotetaan osana prosessia tarkoitukseen sopivia jälkiä analysoitavaksi.

Avainsanat ja -sanonnat: oppimisanalytiikka, tiedonlouhinta, akateeminen analytiikka, korkeakoulututkimus, etiikka, osaamisperustaisuus

1 Johdanto

Korkeakoulutus on muutospaineessa monesta suunnasta. Vähenevillä resursseilla pitää tuottaa entistä enemmän osaamista tietotekniikan mahdollistamilla globaaleilla koulutusmarkkinoilla. Tästä huolimatta korkeakoulutuksen perimmäinen tarkoitus – osaamisen edistäminen ja mahdollisesti sivistäminen koeteltuun tutkimustietoon perustuen – ei ole juurikaan muuttunut. Nopeasti kehittyvässä maailmassa faktatietojen ulkoa opettelun sijaan koulutuksen painopiste on siirtynyt yleisten ja alakohtaisten osaamisten tuottamiseen ja osoittamiseen. Korkeakoulut ovat alkaneet jäsentää toimintaansa annettavan opetuksen sijaan oppijalle tuotetun osaamisen näkökulmasta.

Yksi korkeakoulutuksen muutospaineen syy ja samalla potentiaalinen ratkaisu on tietotekniikka, jonka nopea kehitys on mahdollistanut opiskelun siirtymisen paikasta ja ajasta riippumattomiin digitaalisiin oppimisympäristöihin. Opiskelusta jää näissä digitaalisissa oppimisympäristöissä jälkiä, joita voidaan

tutkia, analysoida ja hyödyntää erilaisiin tarkoituksiin, esimerkiksi oppimisen edistämiseksi.

Koulutuksen ohella muillakin elämän osa-alueilla digitaalisuus sulautuu osaksi työtä ja arkea. Erilaiset järjestelmät keräävät tallentavat dataa toiminnastamme. Datan määrän räjähdysmäisen kasvun myötä tiedon analytiikkaan, louhintaan ja koneoppimiseen liittyvät kysymykset ovat tulleet ajankohtaisiksi tietojenkäsittelytieteessä. Datan jalostamisen ja hyödyntämisen yleistymisen myötä myös tiedon käyttöön liittyvät eettiset kysymykset ovat herättäneet keskustelua.

Datan jalostaminen tiedoksi ja tiedon hyödyntäminen päätöksenteossa ovat mahdollisuus, jota kohtaan on suuria odotuksia myös korkeakoulutuksessa. Aiheen ympärille on muodostunut, jos nyt ei aivan uusia tieteenaloja, niin ainakin jonkinlaisia monitieteisiä tutkimussuuntauksia, joista keskeisimpiä ovat oppimisanalytiikka (learning analytics, LA), koulutustiedon louhinta (educational data mining, EDM), akateeminen analytiikka (academic analytics, AA) sekä korkeakoulututkimus (institutional research, IR).

Esittelen yleisellä tasolla, mitä näillä tutkimussuuntauksilla tarkoitetaan ja pohdin sitä, miten kyseiset alat liittyvät tietojenkäsittelytieteeseen. Tästä syvennän tarkastelua käyttämällä esimerkkinä osaamisperustaisen arvioinnin tukemiseen kehitettyä sovellusta ja sovelluksesta tehtyä tutkimusta. Esittelen ja analysoin tämän esimerkin valossa siinä käytettyjä teknisiä menetelmiä ja niiden eettisiä ulottuvuuksia. Perustelen tätä lähestymistapaa sillä, että käsitykseni mukaan eettinen tarkastelu on hedelmällisimmillään käydessään loputonta vuoropuhelua teorian ja empirian välillä. Samalla pyrin osoittamaan, että vallitsevat käsitykset erityisesti oppimisanalytiikan vakiintumassa olevasta tehtäväkentästä eivät ole riittäviä vastatakseen korkeakoulutuksen muutospaineeseen erityisesti osaamisperustaisuuden korostumiseen.

2 Korkeakoulussa tuotetun datan hyödyntämisen suuntaukset

Korkeakoulutukseen liittyvän datan hyödyntämiseen on muodostunut useita tutkimussuuntauksia, joiden keskeisimpien piirteiden esittelyn tarkoituksena on seuraavaksi luoda akateeminen konteksti tulevalle osaamisanalytiikkaan liittyvälle tarkastelulle sekä kuvata näiden suuntausten yhteyttä tietojenkäsittelytieteeseen.

Oppimisanalytiikan ymmärrän yleiskäsitteeksi toiminnalle, jonka tarkoitus on useimmiten opiskelijatietojärjestelmien (student information system) ja oppimisen hallintajärjestelmien (learning management system) tietojen avulla lisätä ymmärrystä oppimisesta ilmiönä. Toisin sanoen, Gaševićin ja muiden [2015, 64] mukaan, oppimisanalytiikkaa on ”datan mittaaminen, kerääminen, analysointi

ja raportointi oppijoista ja oppimisympäristöistä tarkoituksena ymmärtää ja optimoida oppimista ja oppimisympäristöjä." Erityisenä tavoitteena oppimisanalytiikassa on pidetty pyrkimystä luoda vuoropuhelua tietojenkäsittelytieteilijöiden ja koulutuksen tutkijoiden välille. [Siemens and Baker 2012, 252.] Gaševićin ja muiden [2015, 65] tutkimuksessa oppimisanalytiikan yleisimmiksi tavoitteiksi on havaittu opiskelumenestyksen ennakoiminen ja ennakoivan palautteen antaminen oppijalle. Näiden aiheiden lisäksi oppimisanalytiikan käsitteen alla tehdään hyvinkin moninaista työtä erilaisine lähestymistapoineen, menetelmineen ja tavoitteineen. Moninaisuudesta huolimatta oppimisanalytiikan tavoitteena on tämän perusteella useammin opiskelemisen käytännön tukeminen kuin teoreettista ymmärryksen lisääminen.

On huomion arvoista, että oppimisanalytiikka on hieman harhaanjohtava termi sille, mitä käytännössä oppimisanalytiikassa tehdään. Yhteistä useimmille tutkimuksille on se, että niissä hyödynnetään jotakin opiskelijan toimintaa kuvaavia olemassa olevia tietoaineistoja. Hyödynnettävien aineistojen, esim. erilaisten loki- ja rekisteritietojen, perusteella kyse vaikuttaisi olevan ennemminkin opiskeluanalytiikasta, jolloin tarkastellaan opiskeluun liittyviä tapahtumia eikä sinällään suoranaisesti oppimista tai osaamisen kehittymistä. Tällöin tehdään oletus, että opiskelukäyttäytymisestä jäävät digitaaliset jäljet kertovat jotakin myös oppimisesta tai osaamisen kehittymisestä. Mielestäni tämän oletuksen oikeellisuuden arviointi on keskeistä oppimisanalytiikkatoteutusten ja -tutkimuksien arvioinnissa.

Oppimisanalytiikan ohella toinen varteenotettava, mutta jonkin verran oppimisanalytiikan tekijöistä erillinen, tutkimussuuntaus on koulutustiedon louhinta. EDM määritellään "syntyvänä tieteenalana, jossa kehitetään menetelmiä koulutuskontekstille ominaisen datan tarkasteluun ja näiden menetelmien käyttöön tarkoituksena paremmin ymmärtää opiskelijoita ja oppimisympäristöjä." [Siemens and Baker 2012, 252.] Oppimisanalytiikalla ja EDM:llä on yhteisiä tavoitteita ja kiinnostuksen kohteita, mutta toisistaan poikkeavia teknologisia, ideologisia ja metodologisia suuntauksia. Rajanveto näiden kahden tiedeyhteisön välille on jossakin määrin hankalaa, mutta hieman yleistäen voidaan Siemensin ja Bakerin [ibid.] tavoin todeta, että tiedonlouhinta on enemmän teknologialähtöistä sekä menetelmiin keskittyvää oppimisanalyytikoiden painottaessa enemmän opettajan työn ja opiskelun tukemista sekä teknologian instrumentaalista roolia oppimisprosessissa.

Aihetta ulkopuolelta tarkastelevalle näyttääkin siltä, että tiedonlouhijat ovat ensisijassa tietojenkäsittelytieteilijöitä, jotka soveltavat osaamistaan koulutukseen liittyvän tiedonlouhinnan osa-alueeseen oppimisanalyytikoiden ollessa kasvatustieteilijöitä, jotka haluavat hyödyntää tietojenkäsittelyn menetelmiä

omassa kiinnostuksen kohteessaan. Toki tämäkin jaottelu on varmasti karkea yleistys siitä, millaisilla taustoilla ja intresseillä yksittäiset tutkijat liikkuvat näissä tiedeyhteisöissä ja niiden välillä.

Kolmas aiheeseen keskeisesti liittyvä käsite on akateeminen analytiikka. Akateemisella analytiikalla tarkastellaan kahden edellisen suuntauksen tavoin opiskelijoiden käyttäytymistä. Keskeisin ero vaikuttaisi olevan se, että tiedon julkilausuttu käyttötarkoitus akateemisessa analytiikassa on korkeakoulun päätöksenteon ohjaaminen ja sen myötä korkeakoulun toiminnan, eli usein nykyisin käytännössä kilpailukyvyn, kehittäminen. Näin ollen akateemisessa analytiikassa on enemmän kyse liiketoimintatiedon hallinnasta ja analytiikasta (business intelligence, business analytics) korkeakoulun kontekstissa kuin varsinaisesta tutkimusalasta. Akateemisessa analytiikassa eniten tarkasteltuja aiheita ovat uusien opiskelijoiden rekrytoinnin ohella koulutuksen läpäisy ja keskeyttäminen. Aihealueet ovat valikoituneet siksi, että näihin asioihin vaikuttamalla voidaan nähdä mittaviakin taloudellisia hyötyjä yliopistolle. [Campbell and Oblinger 2007, 2.]

Siinä missä oppimisanalytiikan ja EDM:n tiedonintressi on ensisijassa välineellinen kehiteltäessä teknisiä sovelluksia opiskelun ja opettamisen tueksi, liittyy akateemisen analytiikan tiedonintressi enemmän koulutusorganisaation tiedolla johtamiseen. Erona perinteiseen liiketoiminta tiedon esittämiseen Campbell ja Oblinger [2007] esittävät sen, että akateemisessa analytiikassa tavoitteena on usein mennä erilaisten raporttien ja visuaalisten mittaristojen (dashboard) esittämistä syvemmälle tiedon louhinnan ja predikatiivisen tilastotieteen menetelmien avulla.

Viimeisestä, kolmelle edeltävälle suuntaukselle läheisestä, käytän paremman nimen puutteessa käsitettä korkeakoulututkimus, jolla tarkoitan englanninkielisen institutional research (IR)-käsitteen alla tehtävää työtä. Edellisten suuntausten tavoin IR on järjestäytynyt omiksi järjestöikseen ja julkaisuikseen. Reichard [2012, 3] kuvaa IR:n kehittyneen 1960-luvulla yksittäisten korkeakoulujen tekemästä oman toimintansa tutkimuksesta. 1960-luvulta lähtien aihetta varten perustetut IR-osastot tai -yksiköt alkoivat toteuttaa tätä tehtävää amerikkalaisissa yliopistoissa. IR:ssä kyse on korkeakoulun operatiivisen toiminnan tukemisesta teoreettiseen ymmärrykseen pohjautuen. Useimmiten alan toimijat edustavat korkeakoulutuksen hallintoa. Teoksessa Handbook of Institutional Research [2002, xix] on lainattu Joe Saupen määritelmää IR:stä: ”IR on korkeakoulutusorganisaation sisäistä tutkimusta, joka tuottaa tietoa organisaation suunnittelun, johtamisen ja päätöksenteon tueksi.” Määritelmään sisältyy kyseisen käsikirjan mukaan niin opettamisen kuin opetussuunnitelman tutkimus,

opiskelijavalintojen, arvioinnin ja työkuormaan liittyvän ymmärryksen lisääminen, korkeakoulutoimijoiden roolien ymmärtäminen, koulutuksen kulujen tarkastelu sekä koulutusorganisaation toiminnan kuvailu tilastojen ja aikasarjojen muodossa. Teknisinä lähestymistapoina voidaan edellä mainitun käsikirjan esimerkkien valossa nähdä aivan vastaavia kuin mitä oppimisanalytiikan ja EDM:n parissa toimivat hyödyntävät, mukaan lukien erinäiset tiedonlouhinnan ja tilastotieteen menetelmät. Toiminnan tavoitteiden näkökulmasta IR:ssä on kyse hyvinkin samankaltaisesta toiminnasta, kuin mitä tehdään akateemisen analytiikan alla.

Tässä tarkastelemistani eri suuntauksista keskeisimpänä yhdistävänä tekijänä on korkeakoulutuksessa syntyvän tiedon hyödyntäminen eri tarkoituksiin. Suuntaukset eivät ole mitenkään selkeästi rajattuja vaan samankaltaista ja samaan tarkoitukseen tehtävää toimintaa voidaan tehdä oikeastaan kaikkien suuntausten alla. Keskeisimpänä erottelevana tekijänä tuntuu ennemminkin olevan se, kuka tutkimusta tekee: koulutuksen hallinnoitsijat (erityisesti AA ja IR), tietojenkäsittelytieteilijät (erityisesti EDM), opettajat tai kasvatustieteilijät (erityisesti LA). Toinen erottava tekijä on se, kenen ajatellaan ensisijassa hyötyvän työstä: erityisesti LA:ssa, mutta myös EDM:ssä tavataan painottaa opiskelijan näkökulmaa IR:n ja AA:n painottaessa korkeakoulutuksen järjestäjän näkökulmaan.

Jo tämän esittämäni pintapuolisen tarkastelun avullakin voidaan todeta, että oppimista, opiskelua ja korkeakoulutusta yleensä voidaan tarkastella hyvin moninaisin tavoin ja tavoittein. Tämän tarkastelun kannalta keskeistä on se, että riippumatta siitä, minkä tutkimussuuntauksen alla ja kenen toimesta tutkitaan, tietojenkäsittelytiede muodostaa tälle toiminnalle ennen kaikkea teknisen ja menetelmällisen varannon. Kaikissa edellä mainituissa suuntauksissa kyse on tavalla tai toisella tietotekniikan hyödyntämisestä korkeakoulutusta koskevan ymmärryksen lisäämisessä. Tällöin keskeisiä ovat tietojenkäsittelytieteen erinäiset tiedon hallintaan, tallentamiseen, analysointiin ja esittämiseen liittyvät teoriat, menetelmät ja käytännön sovellukset, jotka ovat pitkälti yhteneviä rippumatta siitä, millä substanssialueella niitä sovelletaan. Suurimpana erona muihin substanssialueisiin, esimerkiksi liike-elämään tai julkishallintoon yleisesti tai vaikkapa terveydenhoitoon verrattuna, en näe niinkään sitä, että korkeakoulutus edellyttäisi erityisiä muista aloista poikkeavia tietojenkäsittelytieteellisiä ratkaisuja. Sen sijaan ennemminkin kyse on siitä, että tietotekniikan soveltaminen korkeakoulutuksessa edellyttää onnistuakseen välttämättä jonkinasteista pedagogista asiantuntemusta.

Tämän yleisen akateemisen tarkastelun myötä on luontevaa jatkaa esimerkkitoteutuksella arvioinnilla. Käytän jatkossa oppimisanalytiikan käsitettä yleiskäsitteenä korkeakoulutusta koskevan tiedon tutkimukselle ja hyödyntämiselle. Oppimisanalytiikasta haluan kuitenkin tietoisesti erottaa osaamisanalytiikaksi kutsumani toiminnan, josta seuraavassa kuvaamani ja arvioimani toteutus on esimerkkitapaus. Osaamisanalytiikan määrittelen toimintana, jossa oppimisanalytiikkaa toteutetaan erityisesti osaamisperusteisesta näkökulmasta, eli korkeakouluopiskelijan osaamisen kehittämisessä ja arvioinnissa tai näitä tukevassa toiminnassa.

3 SCALA-osaamisanalytiikkasovelluksen tekninen toteutus

Tarkastelen tässä luvussa Rayón ja muiden [2014 ja 2015] tutkimuksia, joissa on kuvattu osaamisperustaisen arvioinnin tukemiseen kehitettyä SCALA-sovellusta. Tutkimukset ovat hyviä esimerkkejä niistä harvoista osaamisanalytiikkaa käsittelevistä artikkeleista, joita viime vuosina on kirjoitettu. Tarkastelun avulla nostan erityisesti esiin osaamisanalytiikaksi kutsumaani näkökulmaa sekä osaamisanalytiikan toteutuksessa käytettyjä tietojenkäsittelytiedettä sivuavia menetelmiä. Tähän tarkoitukseen artikkelit sopivatkin erinomaisesti, sillä SCALAssa on käytetty poikkeuksellisen kattavasti erilaisia menetelmiä.

3.1 SCALAlla tukea osaamisen arviointiin

Rayónin ja muiden tavoitteena on ollut rakentaa työkalu, joka tukee opettajan työtä osaamisen arvioinnissa ja opiskelijaa osaamisen kehittymisessä. Osaamisperustaisessa koulutuksessa keskeistä on opetussuunnitelmaan kuvatut osaamisen kriteerit (rubrics), joita vasten oppijan kehittymistä arvioidaan. Osaamisperustaisessa opetussuunnitelmassa kuvataan osaamiskriteereillä eri osaamisen tasoilla se, millaisilla tekijöillä osaamisen taso määritellään. Osaaminen voidaan mieltää oppialakohtaisina sisällöllisinä osaamisina, esimerkiksi tietojenkäsittelytieteessä ohjelmointiosaaminen, tietokantaosaaminen tai käyttöliittymäsuunnittelu. Geneerisillä kompetensseilla tarkoitetaan oppialasta riippumattomina yleisiä osaamisia, esim. ryhmätyötaidot, projektinhallintataidot, neuvottelutaidot, jne. Aikaisemmassa artikkelissaan Rayón ja muut [2014] keskittyvät geneerisiin osaamisiin ja tarkalleen ottaen niistäkin ainoastaan yhden, ryhmätyötaitojen, kehittymisen arvioinnin tukemiseen. Myöhemmin ilmestyneen artikkelin [Rayon ja muut, 2015] näkökulma on yleisempi, vaikka siinä ei suoranaisesti avatakaan sitä, miten käytännössä sovelluksen laajentaminen ryhmätyötaidoista yleisem-

mälle tasolle toteutetaan. Ryhmätyötaitojen kehittymisen kyseiset tutkijat olettavat olevan seurausta opiskelijoiden vuorovaikutuksesta sähköisissä oppimisympäristöissä, jolloin oletuksena on, että keräämällä määrällistä tietoa vuorovaikutuksesta ja tätä tietoa analysoimalla voitaisiin sanoa jotakin ryhmätyötaitojen kehittymisestä. Tätä olettamusta pidän tutkimuksen yhtenä heikkona kohtana – vuorovaikutuksen määrästä ei suoraan voi päätellä siitä, että ryhmätyötaidot olisivat kehittyneet.

Yleisesti ottaen geneeristen kompetenssien arviointi on haasteellista, mutta Rayónilla ja muilla [2014, 291] on näkemys siitä, että osaamisanalytiikan avulla voidaan tukea osaamisperustaista arviointia yhdistämällä tietoa useista lähteistä ja käyttämällä tiedon analysointiin monipuolisia menetelmiä. Tällöin Rayón ja muut puhuvat oppimisanalytiikan avulla rikastetuista osaamistavoiteitteista (enriched rubrics). Perinteiset kvantitatiiviisin lokien tulkintaan ja esim. tenttiarvosanoihin perustuvat oppimisanalytiikka sovellukset eivät Rayónin ja muiden [2014, 292] mukaan ole soveltuvia kompetenssien arviointiin, koska ne eivät anna käyttökelpoista informaatiota osaamistavoitteiden eritasoisista osaamisista. Kyseiset tutkijat eivät olleet löytäneet työkalua tai arkkitehtuuria, joka olisi tarkoitettu erityisesti osaamisen kehittymisen seuraamiseen korkeakoulutuksessa. Tästä syystä he ovat kehittäneet SCALA-sovelluksen, jonka tarkoitus on avustaa opettajaa kokonaisvaltaisesti lähestymään sekä oppimisen näytteitä (products) että monimuotoista oppimisprosessia.

Ryhmätyötaidot on ollut looginen valinta osaamisanalytiikan sovellukseksi – on mahdollista tehdä jokseenkin uskottava oletus, että mitattavissa oleva vuorovaikutus sähköisissä oppimisympäristössä kuvaa jollakin tavalla ryhmätyötaitojen kehittymistä. Kokonaan toinen kysymys ja huomattavasti enemmän mielikuvitusta vaativa on se, että mistä datasta ja millä tavoin mitattaisiin muiden osaamisten kehittymistä.

3.2 SCALAn rakenne

Rayón ja muut ovat tuotteistaneet osaamisanalytiikkaratkaisunsa SCALA-työkaluksi. SCALAn toteuttamisessa on käytetty Caliper-arkkitehtuuria[1]. Caliper on oppimisanalytiikkastandardi, jonka tarkoitus on standardisoida korkeakou-

[1] https://www.imsglobal.org/activity/caliperram

lutusta koskevan tiedon esittämistä ja hyödyntämistä. Caliper-arkkitehtuuria pidetään parempana kompetenssien arvioinnin tukemisessa kuin vastaavaa, ja ehkä nykyisin tunnetumpaakin, Experince API (xAPI)-määrittelyä[2]. [ibid., 294.]

SCALA koostuu neljästä osasta: tietolähteet, tiedon integrointi, tietovarasto ja visuaalinen mittaristo (dashboard). Tietolähteenä SCALAssa on joukko opiskelijoiden kanssa ennalta sovittuja webteknologioita, joista dataa on kerätty eri muodoissa. Opiskelijoilla on ollut käytössä Moodle ja MediaWiki (data sql-tietokannoissa), Google Docs ja Google Forms-palvelut (data rakenteettomana JSONina), Youtube (data csv-tiedostona) sekä Twitter (data rakenteisena JSONina). Tiedon integrointia varten on rakennettu palveluväyläratkaisu (Service Bus), joka kerää ja muuntaa tietoa eri lähdejärjestelmistä ja siirtää SCALA-järjestelmään. ETL-prosessin (Extract – Transform – Load) tekninen toteutus on tehty Pentaho Kettle -järjestelmän Data Adapter-työkalulla. SCALA-järjestelmässä tietokantanaratkaisuna on käytetty MongoDB:tä. Visuaalinen mittaristo on toteutettu Highcharts-nimisellä Javascript-kirjastolla, joka on tarkoitettu interaktiivisten kaavioiden luontiin web-sivuille. Visuaalinen mittaristo on jaettu kolmeen osaan: kaavioihin, taulukoihin ja analytiikkaan. [ibid., 295.]

SCALAn toteuttajien tarkoitus ei ole ollut vain visualisoida opiskeludataa vaan louhia sitä tavoitteena löytää erilaisia opiskelun malleja. Tällaiseen tiedon analysointiin SCALAssa on käytetty klusterointiin ja assosiaatiosääntöihin (M5-algoritmi) perustuvia tiedonlouhinnan menetelmiä. Tiedonlouhinta on toteutettu Pentahon Weka-työkalulla. K-means-klusterointialgoritmilla sekä assosiaatiosäännöillä on etsitty tietokannan attribuuttien välisiä yhteyksiä. Tiedonlouhintamenetelmillä tuotetun tiedon visualisointia ei artikkelin kirjoitushetkellä ollut toteutettu. [ibid., 296]

SCALAssa on toteutettu käytännössä tietovarasto opiskelua koskevasta datasta ja analysoitu toteutetun tietovaraston sisältöä. Varsinaisen osaamisanalytiikan kohteena on ollut ryhmätyötaitoja koskevan osaamisen arviointi. Tehdyn kuvauksen perusteella vaikuttaa siltä, että tehty on ikään kuin esimerkkitoteutus siitä, että kuvatun kaltainen on ylipäänsä mahdollista. Toteutukseen voitaisiin tarvittaessa kytkeä muita tietolähteitä toteuttamalla ETL-prosessi näistä tietolähteistä. Siitä huolimatta toteutuksen esimerkinomaisuus ja skaalautuvuus herättävät kysymyksiä sekä teknisesti että myös erityisesti sen osalta, miten muita yleisiä tai alakohtaisia osaamisia olisi mahdollista arvioida ratkaisun avulla ja se suhteen, että mitä SCALAssa tällä tavoin laajennettaessa käytettäisiin tietolähteinä.

---

[2] https://www.adlnet.gov/adl-research/performance-tracking-analysis/experience-api/).

3.3 K-means klusteroinnista

K-means-klusterointi on ohjaamattoman (unsupervised) koneoppimisen menetelmä, eli siinä on kyse kuvailevasta mallinnuksesta, jossa ei pyritä ennustamiseen. Menetelmää on yleisesti käytetty ryhmittelemään kohteita vastaavien kohteiden kanssa samoihin luokkiin. Kyseistä menetelmää käytetään laajalti, ja oppimisanalytiikassakin sitä on käytetty esimerkiksi samankaltaisten oppimisten tapojen etsimiseen opiskelijaryhmistä tai ryhmätyöhön perustuvaan opiskeluun kannustamiseen. [ibid., 296; Vellido *et al.* 2011, 78.]

K-means-klusterointi vastaa ihmismielen taipumusta ryhmitellä tietoa luokkiin. Menetelmän avulla on mahdollista ryhmitellä dataa siten, että menetelmän käyttäjä ei etukäteen nimeä luokkia vaan mahdollinen nimeäminen tapahtuu jälkikäteen tuloksesta riippuen. [Vellido *et al.* 2011, 76] Tilastotieteen näkökulmasta on kyse monimuuttujamenetelmästä.

Klusterointimentelmät on tapana jakaa hierarkkisiin ja osittaviin. Hierarkkisessa klusteroinnissa oletetaan, että klusterin rakenne esiintyy eri sisäkkäisillä tasoilla. Tätä määritelmää tarkentaa osittava klusterointi, jota k-means-algoritmikin edustaa. Osittavassa klusteroinnissa oletetaan monen sisäkkäisen tason sijaan yksi yhteinen taso kaikille klustereille. K-means-klusterointia voidaan kuvata formaalimmin siten, että menetelmän tavoitteena on jakaa data $D = \{x_1, x_2, \dots, x_n\}$ K määrään erillisiä ryhmiä $C = \{C_1, C_2, \dots, C_K\}$, joissa jokainen x_i on asetettu johonkin ryhmään C_K. [Vellido *et al.*, 77-79]

K-means algoritmi toimii neljässä vaiheessa:

1) valitse satunnaiset K pistettä ryhmien keskuksiksi,
2) aseta kaikki havainnot lähimpään ryhmäkeskukseen samankaltaisuuden perusteella,
3) laske uudelleen ryhmien keskukset ja
4) toista kohtia 2) ja 3) niin kauan, että ryhmien jäsenyydet vakiintuvat.

Klusteroinnin tavoitteena SCALA-esimerkissä on se, että opettajat voisivat eriyttää opiskelijoiden ohjausta klusteriin kuulumisen perusteella ja näin henkilökohtaistaa opetusta ja palautetta osaamisen kehittymisestä. Keskeisenä menetelmän käytössä pidetään käyttäjän tekemään valintaa klustereiden määrästä – lopputulos riippuu paljon ryhmäkeskusten alkuperäisestä tilanteesta. SCALAssa klustereiden maksimimääräksi oli määritelty viisi, jolloin tuloksena on saatu viisi yhteisten piirteiden mukaista ryhmää. [Rayón *et al.* 2014, 296-7.] Toinen käyttäjästä riippuva tekijä on samankaltaisuuden mittaamisessa käytettävä laskentatapa. SCALA-esimerkissä oli käytetty yleisintä tapaa, euklidista etäisyyttä, joka tuottaa pallomaisia klustereita.

Vellidon ja muiden [2001, 77] mukaan klusterointimenetelmän validointi on haastavaa, koska tulkinta ryhmittelyn tarkoituksenmukaisuudesta tai kiinnostavuudesta on subjektiivinen. Tämän subjektiivisen ulottuvuuden ovat SCALAa arvioineet tutkijat huomioineet korostaessaan tiedon hyödyntäjän roolia tiedon tulkitsijana. [Rayon *et al.* 2014, 297] Tutkijat eivät sitä vastoin ole avanneet sitä, millaista ymmärrystä käytetystä menetelmästä ja sen sisältämistä ennakko-oletuksesta tulkitsijalta edellytetään.

3.4 Tiedonlouhinnasta assosiaatiosäännöllä

Ostoskorianalyysinä usein sovelletussa assosiaatiosääntömenetelmällä selvitetään sitä, mitkä tarkasteltavista tapahtumista liittyvät toisiinsa. Vähittäismyyntiä esimerkkinä käytettäessä voidaan esimerkiksi havaita, että ostoskorin sisältöä analysoimalla ”70% asiakkaista, jotka ostavat viiniä ja juustoa, ostavat myös viinirypäleitä”. [Höppner 2010, 299.] Tuloksen perusteella voidaan suunnitella myynnin lisäämiseen tähtääviä toimenpiteitä. Korkeakoulutukseen sovellettuna voitaisiin tarkastella, mitä asioita opiskelija tekee sähköisissä oppimisympäristöissä suhteessa siihen, miten paljon opiskelija on oppinut: esimerkiksi ”70% niistä, jotka ovat katsoneet kaikki opetusvideot ja tehneet kaikki harjoitustehtävät, saavat kurssista erinomaisen arvosanan”. Tämän tiedon perusteella opettajan olisi mahdollista muokata kurssin toteutusta ja esimerkiksi motivoida opiskelijoita.

Toinen Rayónin ja muiden tutkimuksessa käytetty tiedonlouhinnan menetelmä on ohjattu (supervised) assosiaatiosääntömenetelmä, jossa on käytetty Quinlanin [1992] M5-algoritmia. SCALAssa riippuvana muuttujana oli loppuarvosana, jonka opettaja antanut arvioituaan oppimista SCALAn tietojen perusteella. Riippumattomana muuttujana oli käytetty debattien lukumäärää ja osallistumista, testitulosta, Google Docsissa reflektioon käytettyä aikaa ja reflektointikertojen määrää. Tällä menetelmällä Wekan avulla on löydetty kymmenen karsimatonta (unpruned) regressiosääntöä. [Rayón *et al.* 2014, 297.]

Quinlanin [1992, 343] mukaan M5-algoritmin tarkoitus on arvoja ennustavan oppivan mallin luominen päätöspuuhun perustuvan menetelmän avulla. Erotuksena regressiopuihin, joissa arvot sijaitsevat puun lehdissä, M5-algoritmin puissa voi olla monimuuttujaisia lineaarisia malleja. Näin M5:n etuna Quinlanin [ibid., 348] mukaan on se, että siinä voidaan tehokkaasti ja tarkasti ennustaen käsitellä satoja attribuutteja toisin kuin monissa muissa vastaavissa menetelmissä, joissa suuri määrä attribuutteja edellyttää laskentatehon lisäämistä. Menetelmän tarkoituksena on löytää attribuuttien välisiä suhteita (muodossa: jos

attribuutin A (riippumaton muuttuja) arvo on X, niin silloin attribuutin B (riippuva muuttuja) arvo on Y). Näitä attribuuttien välisiä suhteita SCALAn esimerkissä löytyi kymmenen kappaletta.

M5-menetelmän löytämät kymmenen sääntöä jätetään tutkimuksessa tarkemmin analysoimatta. Assosiaatiosääntömenetelmien yleisenä haasteena on menetelmän parametrien määrittelyn ohella menetelmän tuottamisen sääntöjen suuri määrä. [García *et al.* 2011, 97-8.] Tutkimuksessa ei avata M5-mentelmässä käytettyjä parametrejä, menetelmän tuottamien sääntöjen merkitystä eikä analysoida tarkemmin sitä, mitä säännöistä voidaan pitää relevantteina. Päätöspuun karsiminen tai erinäisten tilastotieteellisten menetelmien käyttäminen relevanttien sääntöjen löytämiseksi olisi ollut mahdollista. [ibid.]

Tutkimuksessa käytettiin riippuvana muuttujana kurssin loppuarvosanaa, joka on perustunut opettajan harkintaan SCALAn tuottaman tiedon ohjaamana. Menetelmällä saadut kymmenen sääntöä kuvaavat sitä, miten riippumattomiksi attribuuteiksi valitut muuttujat selittävät loppuarvosanaa. Tässä piilee riski siitä, että M5-menetelmän tulos kuvaa enemmin opettajan arviointitapaa kuin opiskelijan toimintaa sinänsä.

Tutkimuksesta ei selviä perustetta M5-algoritmin valinnalle. Koska tutkimuksessa tarkasteltiin vain harvalukuista muuttujien määrä, olisi ainakin laskentatehovaatimuksen näkökulmasta menetelmäksi voitu valita muukin menetelmä.

4 Eettistä osaamisanalytiikkaa SCALA-sovelluksella

Tässä luvussa avaan sitä, millaisia eettisiä haasteita liittyy oppimisanalytiikkaan. Etiikka on ollut aihe, jota on painottunut viime vuosina paljon alan konferenssiessa. [Drachsler *et al.* 2015.] Seuraavassa luomani yleisen kuvauksen perusteella arvioin, mitkä eettiset näkökohdat ovat relevantteja edellä esittelemässäni osaamisanalytiikkaa koskevassa toteutuksessa.

Etiikkaa voi lähestyä sekä soveltamalla yleisen tason periaatteita yksittäiseen tapaukseen tai päinvastoin johtamalla yksittäisestä tapauksesta mahdollisia yleisiä periaatteita. Parhaimmillaan eettinen tarkastelu voi olla vuoropuhelua yksittäisen ja yleisen välillä. Tämän takia pidän käytännön esimerkin tarkastelua olennaisena.

Oppimisanalytiikan monimuotoisuus johtaa helposti siihen, että yleinen oppimisanalytiikan eettinen säännöstö jää välttämättä tarpeettoman yleiselle tasolle, jolloin ei ole olennaista eroa siihen, että oppimisanalytiikan alalla sovellettaisiin yleisiä tutkimus- ja informaatioetiikkaan liittyviä eettisiä periaatteita ja

normeja, ks. esim. [Moore 2005]. Vastoin tätä käsitystäni erityisestä oppimisanalytiikan etiikan tarpeettomuudesta, erilaisia yleisiä oppimisanalytiikan eettisiä viitekehyksiä on viime vuosina esitelty lukuisia.

4.1 Sladen ja Prinsloon eettiset periaatteet

Oppimisanalytiikan toteuttamistapoja ja käyttötarkoituksia on lukuisia. Lisäksi oppimisanalytiikan soveltajilla on erilaisia ideologisia ja epistemologisia olettamuksia. Tämän vuoksi ei välttämättä ole hedelmällistä yrittää muodostaa yleispätevää eettistä ohjeistoa oppimisanalytiikalle. Tästä huolimatta Slade ja Prinsloo [2013, 1519-21] ehdottavat joukon yleisiä periaatteita, joiden avulla koulutusorganisaation on mahdollista rakentaa omaan tilanteeseensa sopiva eettinen ohjeistus. Periaatteet ovat seuraavat:

1) oppimisen analytiikka tulee nähdä moraalisena toimintana, jonka tarkoitus on enemmin lisätä ymmärrystä kuin mitata toimintaa,
2) opiskelijat tulee nähdä toimijoina ja yhteistyötahona sen sijaan, että opiskelijat nähtäisiin datan tuottajina tai palveluiden ja interventioiden kohteena,
3) opiskelijan identiteetti ja oppiminen tulee käsittää ajallisina ja muuttuvina dynaamisia konstruktioina,
4) opiskelumenestys on kompleksinen ja moniulotteinen ilmiö,
5) korkeakoulujen tulee toimia avoimesti ja kertoa miten ja mihin tarkoituksiin dataa käytetään,
6) korkeakoulutuksessa ei ole varaa olla käyttämättä dataa, koska korkeakoulut ovat joka tapauksessa ovat vastuussa toiminnastaan sidosryhmille, hallitukselle tai opiskelijoille.

SCALAn toteutuksessa tutkijat [Rayón *et al.* 2014, 294] käyttivät ”empiiriseksi validoinniksi” kutsumaansa kolmivaiheista menetelmää. Validointi sisälsi

1) SCALAn kehitykseen osallistuvien opettajien ja opiskelijoiden informoinnin, jolloin kerrottiin myös toteutuksen eettisestä näkökulmasta,
2) opiskelijoiden yksin ja ryhmissä tekemiä oppimistehtäviä käyttäen sovittuja webteknolgioita,
3) opettajalle esitettyjä kaavioita ja taulukoita, joiden tavoitteena oli tukea osaamisperusteista kompentenssien, tässä yhteydessä ryhmätyötaitojen, arviointia; opiskelijoille annettiin mahdollisuus jättäytyä pois tutkimuksesta ja osallistuessaan he hyväksyivät datan säilytyksen ja hallinnan heille kerrotulla tavalla.

Sladen ja Prinsloon esittämistä periaatteista voi SCALAn perusteella todeta, että ensimmäinen periaate, ymmärryksen lisääminen mittaamisen sijaan, on oppimisanalytiikan alaa vahvasti tutkimukseen suuntaava. Niin SCALAssa kuin useissa muissakin oppimisanalytiikkatoteutuksissa keskeinen tavoite on ollut tuottaa tietoa, jonka perusteella voidaan tehdä konkreettisia henkilöön kohdistuvia toimenpiteitä, esimerkiksi opettajat suuntaavat palautteen antamista ja opiskelijat opiskeluaan palautteen perusteella. Vaikka ymmärtämistä ja mittaamista ei ole tarpeen nähdä toisensa poissulkevina, niin SCALAn käytännön toteutuksessa korostuu mittaamisen näkökulma. Lähtökohtaisesti SCALAlla pyritään selvittämään mittaamalla sitä, onko ryhmätyötaitoihin liittyvä osaaminen kehittynyt. Olisi mahdollista, että oppimisanalytiikan alalla yleisesti painotettaisiin ymmärtämisen näkökulmaa, jolloin toteutusten tavoitteet olisivat enemmän tieteelliset. Tällöin pyrittäisiin ymmärtämään oppimista ilmiönä. Tieteellisyyttä painotettaessa heräisi tosin kysymys siitä, miten näin ajateltuna oppimisanalytiikka eroaa yleisestä kasvatustieteellisestä tutkimuksesta.

Käytännössä nykyisin oppimisanalytiikkaa tehdään tavoitteena mitata ja saada tietoa käytännön toimenpiteiden perusteeksi, mutta myös toisaalta hyvinkin kunnianhimoista tieteelliseen ymmärrykseen tähtäävää tutkimustyötä. Tässä tarkastelussa kuitenkin olennaista on se, että toisen näkökulman preferoiminen toisen sijaan ei ole eettisessä tarkastelussa relevanttia. Kumpaa tahansa voidaan tehdä eettisesti tai epäeettisesti. Näin ollen ymmärryksen preferoimisessa, vaikuttaa olevan kyse enemminkin oppimisanalytiikan alan määrittelystä ja suuntaamisesta kuin etiikasta sinänsä. Oppimisanalytiikan alan määrittelyn tarve on kasvava Conden ja muiden [2015, 261] tekemien alan moninaistumisesta kertovan huomioiden perusteella.

Voidaan arvioida, että SCALAssa viisi muuta edellä mainittua eettistä periaatetta on toteutuksesta julkaistujen tutkimusartikkeleiden perusteella otettu huomioon hyvin. Edellä mainittu empiirinen validointi kuvaa mielestäni opiskelijoiden näkemistä yhteistyötahoina (periaate 2). SCALAn tavoite antaa palautetta osaamisen kehittymisestä kurssin suorittamisen aikana kuvaa sitä, että tutkijoilla on tausta-ajatuksena ollut oppijan identiteetin ja oppimisen ajallisuus ja muuttuvuus (periaate 3). Osaamisen kehittymisen kompleksisuutta ja monimuotoisuutta (periaate) on SCALAssa pyritty huomioimaan ottamalla mukaan dataa useista lähteistä. Tästä huolimatta voidaan esittää kysymys siitä, onko useista lähteistä kerätyt sähköiset jäljet opiskelusta kuitenkaan riittävä aineisto sille, että SCALAssa tavoitetaan ilmiön kompleksisuuden ymmärtämisen vaade. Osaamisen tunnistaminen ja arviointi ovat aina hyvinkin tulkinnallisia ja arvioijan kokemukseen perustuvia enemmän tai vähemmän subjektiivisia tulkintoja. Toisaalta Rayón ja muut [2014 ja 2015] tuovat esiin sen, että SCALAn tavoitteena

ei ole ollutkaan tuottaa valmista arviota osaamisesta vaan tietoa, joka tukee opettajaa ja opiskelijaa osaamisen kehittämisessä ja arvioinnissa.

Niin ikään toiminnan avoimuus (periaate 5) oli huomioitu sekä toteutuksessa kommunikoitaessa siitä opettajille ja opiskelijoille että sen myötä, että toteutusta on lähestytty myös tutkimuksellisin keinoin, jolloin toteutus saatettu kansainvälisen alan tutkimusyhteisön arvioitavaksi. Toteutus jo olemassaolollaan vastaa korkeakoulutuksen datan hyödyntämisen (periaate 6) vaateeseen.

Mielestäni nämä kuusi periaatetta tavoittavat ison osan niistä kysymyksistä, joita oppimisanalytiikassa tulee ratkaista. Slade ja Prinsloo [2013, 1521] tuovat periaatteiden lisäksi esiin mielestäni hyvinkin keskeisen näkökulman; eettisyyttä arvioitaessa tulee heidän mukaansa aina pohtia sitä, kuka oppimisanalytiikasta hyötyy ja miten? SCALAssa hyötyjiksi on mainittu sekä opettajat että opiskelijat. [Rayón *et al.* 2014, 293.] Jos lähtökohtana on se, että opiskelijat omistavat korkeakoulutuksessa tuottamansa datan, on mielestäni perusteltua eettisestä näkökulmasta, että opiskelijat ovat myös dataa käytettäessä hyödyn saajia. Tämä sillä varauksella, että opiskelijat eivät syystä tai toisesta tietoisesti luovuta dataa käytettäväksi muihin tarkoituksiin. Osaamisanalytiikasta opiskelijat hyötyvät paremman osaamiseensa liittyvän itseymmärryksen kautta joko suoraan tai välillisesti saadessaan kohdistetumpaa palautetta opettajalta.

4.3 LEA's BOX -projektissa laaditut eettiset periaatteet

Euroopan komission rahoittamassa LEA's BOX -projektissa on kehitetty oppimisanalytiikan etiikkaa aihetta käsittelevien tutkimusten sekä kansallisten ja Euroopan unionin lainsäädännön kattavan arvioinnin perusteella. Lopputuloksena on eettinen viitekehys sovellettavaksi projektissa ja projektin myötä jalkautettavissa uusissa toimintatavoissa ja ratkaisuissa. Steiner ja muut [2016, 77] näkevät tämän työn perusteltavana erityisesti siksi, että koulutuksen konteksti, jossa toiminnalla voi olla kauaskantoisia yksilön tulevaisuuteen liittyviä vaikutuksia, tuo mukanaan eettiseen tarkasteluun sellaisia lisäulottuvuuksia, joita yleiset tutkimus- ja informaatioetiikan eettiset tarkastelut eivät saavuta.

Steiner ja muut [ibid., 79-85] nimeävät kahdeksan yksityisyyden ja tiedon suojaan liittyvää eettistä periaatetta. Nämä ovat keskeisiltä piirteiltään seuraavat:

1) Tietosuoja: oppimisanalytiikassa tulee noudattaa lainsäädännöllisiä ja mahdollisia lainsäädäntöä tarkentavia organisaatiokohtaisia sääntöjä sekä tietojärjestelmien suunnitteluun liittyviä hyviä käytäntöjä,

2) tarkoitus ja datan omistajuus: oppimisanalytiikkasovellusten tarkoitus ja tavoitteet tulee olla selkeästi ja ymmärrettävästi rajattu ja jos dataa integroidaan useista järjestelmistä, opiskelijalla on oltava mahdollisuus säädellä järjestelmäkohtaisesti tiedon näkyvyyttä,
3) suostumus: opiskelijoilta tulee pyytää lupa henkilökohtaisen datan käyttöön,
4) avoimuus ja luottamus: opiskelijoille tulee kertoa mitä dataa kerätään ja tallennetaan ja miten sitä analysoidaan,
5) pääsy ja valvonta: opiskelijalle pitää antaa mahdollisuus nähdä ja tarvittaessa korjata kerättyä dataa,
6) vastuullisuus ja arviointi: oppimisanalytiikkasovelluksesta vastaava taho tulee nimetä ja toimintaa tulee arvioida ja kehittää,
7) datan laatu: datan tulee olla edustavaa, relevanttia, oikeellista ja ajantasaista ja
8) tiedonhallinta ja tietoturva: tarpeelliset toimenpiteet tiedon turvaamiseksi luvattomalta tietoon pääsemiseltä, tietojen menetykseltä ja väärinkäytöltä.

Yleisellä tasolla voidaan todeta, että Sladen ja Prinsloon periaatteet ovat yleisempiä ja kokonaisvaltaisempia huomioiden myös toiminnan taustalla olevaa pedagogista kontekstia kun taas LEA's BOX -projektille esitetyt periaatteet ovat astetta konkreettisempia. Näkökulmaerona vaikuttaa olevan se, että Sladen ja Prinsloon periaatteet koskevat enemmän oppimisanalytiikkaa toimintana kun taas LEA's BOX -projektin periaatteet suuntaavat oppimisanalytiikkaan liittyvien tietojärjestelmien käytännön suunnittelua.

LEA's BOX -projektin periaatteista mielestäni erityisesti tietosuojaa ja -turvaa koskevat periaatteet (periaatteet 1 ja 8) laajentavat Sladen ja Prinsloon käsittelyä. Nämä periaatteet ovat luonteeltaan sellaisia, että ne tulee huomioida kaikessa tietojärjestelmäkehityksessä, eivätkä ne siksi ole pelkästään oppimisanalytiikkasovelluksille ominaisia. Tarkastelluissa SCALAa koskevissa artikkeleissa ei suoranaisesti avattu tätä näkökulmaa, mutta ei ole syytä olettaa, etteikö tietoturvaa järjestelmässä olisi huomioitu. LEA's BOX -projektin periaatteiden myötä tietoturvaan liittyvien näkökohtien avaaminen oppimisanalytiikkatoteutusten arvioinnissa on olennaista eikä tietoturvasta huolehtimista tule nähdä itsestään selvyytenä.

Opiskelijoiden mahdollisuus nähdä ja tarvittaessa korjata itseensä koskevaa dataa (periaate 5) sekä järjestelmäkohtaisesti määritellä data käytön oikeuksia (periaate 2) ovat selkeästi Sladen ja Prinsloon periaatteita laajentavia vaateita. SCALA-järjestelmän näkökulmasta näiden vaateiden täyttäminen vaikuttaa eri-

tyisen haastavalta, koska dataa on kerätty paljon ja useista eri lähteistä. Tarkastelujen SCALAa koskevien artikkeleiden perusteella voidaan olettaa, että näitä periaatteita ei ole huomioitu täysin. Kyseiset periaatteet vaikuttavat käytännössä SCALAn tyyppisissä toteutuksissa hyvin haastavilta. Kun on kyse valtavista tietomassoista, miten käytännössä data näytetään omistajalle ja miten annetaan mahdollisuus korjata sitä? Jos opiskelussa käytetään useita eri tietojärjestelmiä, on oikeuksien määrittely järjestelmäkohtaisesti opiskelijallekin työlästä. Onko ylipäänsä SCALAn kaltainen järjestelmä mahdollinen, jos opiskelija haluaa antaa vain tietyn järjestelmän tiedot tähän tarkoitukseen ja estää toisten järjestelmien tietojen käytön? Mielestäni jonkinlainen kokoava opiskelijoilla hyväksytettävä sopimus tietojen käytöstä on käytännön toiminnan kannalta välttämätön.

6 Päätelmät

Olen edellä esitellyt osaamisanalytiikkatoteutuksen sekä arvioinut siihen liittyviä teknisiä ja eettisiä näkökulmia. Lähtökohtana on ollut oppimisanalytiikan tutkimusala, jossa tutkitaan ja kehitetään oppimista tai tarkemmin sanottuna usein opiskelukäyttäytymistä, jonka oletetaan kertovan jotakin oppimisesta. Jos väitetään edelleen, että tällaisen tarkastelun avulla saadaan ymmärrystä osaamisesta, niin jälleen tehdään oletus siitä, että opiskelukäyttäytymisestä voidaan päätellä jotakin osaamisen kehittymisestä.

Olennaista on mielestäni pohtia, voisiko osaamista ja osaamisen kehittymistä lähestyä jotenkin suoremmin. Miten ja millä aineistolla voimme lisätä ymmärrystä erityisesti osaamisen kehittymisestä? Tämä edellyttää välttämättä pedagogisen prosessin tarkastelua ja todennäköisesti sitä, että täyttä vastausta näihin kysymyksiin ei voida antaa oppimisesta jäävien sähköisten jälkien perusteella, vaan jonkinlainen ihmisen arviointiin perustuva elementti osaamisen tunnistamisessa on välttämätön.

SCALA-järjestelmä on vakuuttava ja kunnianhimoinen yritys ratkaista haastavaa tehtävää tietotekniikan keinoin. SCALAssa on käytetty moninaisia teknisiä ratkaisuja, joiden käyttö on avoimesti esitetty kenen tahansa arvioitavaksi. Koska tekniset ratkaisut ovat moninaisia ja laajahkoa alan ymmärrystä edellyttäviä, voidaan esittää kysymys siitä, että onko avoimuus periaatteena riittävä erityisesti oppimisanalytiikan käyttäjän, tässä tapauksessa opettajan ja opiskelijan, näkökulmasta. Pitäisikö käyttäjän myös jollakin tasolla ymmärtää se, mitä eri algoritmeilla on käytännössä tehty, mitä ennakko-oletuksia on tehty ja miten nämä vaikuttavat tulosten tulkintaan? (Ks. esim. [Scantamburlo 2016]) Jos ymmärrystä ei vaadita, niin käyttäjältä odotetaan korkeaa luottamusta järjestelmään ja otetaan riski mahdollisesta järjestelmän perusteella tehtävistä virheellisistä tulkinnoista.

Yleisen oppimisanalytiikkaa koskevan eettisen säännöstön kehittämisen tarpeesta olen eri mieltä useiden sitä puolustavien alan tutkijoiden kanssa. Koska oppimisanalytiikka on yleiskäsite moninaiselle toiminnalle, on eettinen tarkastelu mielestäni perustellumpaa kytkeä tiiviisti yleisempään kuin oppimisanalytiikkakeskeiseen keskusteluun, erityisesti tutkimus- ja informaatioetiikkaan. Oppimisanalytiikassa ei ole tarpeen keksiä pyörää uudestaan. Sitä vastoin konteksti ja tapauskohtaisesti erilaisissa oppimisanalytiikan sovelluksissa on toki huomioitava nämä yleiset eettiset näkökohdat ja niiden soveltaminen koulutuksen alalla.

Oppimisanalytiikkaa koskevien eettisten kysymysten ohella, ja ehkä vielä akuutimmankin tarkastelun tarpeessa, olisivat oppimisanalytiikkatoteutusten ja -tutkimusten pedagogiset taustaoletukset ja se, että miten koulutuksen suunnittelussa voidaan ottaa huomioon jo lähtökohtaisesti myös aineiston tuottaminen oppimisanalytiikalle. Kuten Gašević ja muut [2015, 65] esittävätkin: oppimisanalytiikan haasteena on tulevaisuudessa kytkeytyä nykyistä paremmin oppimista ja opetusta käsittelevään tutkimustietoon. Vaikka eettiset kysymykset isojen datamäärien aikana ovatkin tutkimusetiikallekin haastavia, tutkimustietoon kytkeytyminen voisi tarkoittaa oppimisanalyytikoille myös entistä vahvempaa kytkeytymistä kasvatustieteen tutkimusetiikan pitkään perinteeseen.

Gašević ja muut [2016, 83] toteavat oppimisanalytiikan haasteena yleisemminkin olevan sen, että vaikka on havaittu, että korkeakoulut yleisesti alihyödyntävät dataa ja tämän myötä oppimisanalytiikan sovelluksille olisi suurikin kysyntä, on kuitenkin hyvin riskialtista yrittää soveltaa jonkinlaista yleistä oppimisanalytiikan tekemisen algoritmeja, joka ei ota huomioon korkeakoulu-, ala-, opiskelijapopulaatio- ja opetusmenetelmäkohtaisia eroavaisuuksia. SCALA-järjestelmän näkökulmasta tämä johtaa mielestäni välttämättä järjestelmän skaalautuvuuden tarkasteluun, mitä tarkasteltujen artikkeleiden perusteella ei ole ratkaistu.

Erityisesti LEA's BOX -projektin eettisen periaatteiden myötä eettisyyden arvioinnin yhtenä ulottuvuutena on huomioitava se, että kun korkeakouluilla voidaan nähdä olevan eettinen velvollisuus hyödyntää koulutuksessa tuotettua dataa toiminnan kehittämiseen opiskelijoiden eduksi, etiikkaan ei voida ottaa vain rajoittavaa, esim. yksilönsuojaa käsittävää näkökulmaa, vaan myös eteenpäin vievä ja jopa kehitykseen velvoittava tavoite. Edellä tarkastellut eettiset viitekehykset täydentävätkin toisiaan tästä näkökulmasta.

Käsitelty aihe on laaja ja monisyinen. Näen kuitenkin, että kokonaisuuden hahmottaminen on välttämätöntä, sillä se luo edellytykset monisyisen asian yhteyksien ymmärtämiselle. Samalla näen kuitenkin tarpeelliseksi syventää kutakin aihetta tutkimuksellisesti. Ensimmäinen selkeä jatkoselvittelyn tarve on

osaamisanalytiikan menetelmien ja prosessien kehittäminen ja toinen on syventää käsitystä siitä, miten osaamisanalytiikkaa voitaisiin toteuttaa skaalautuvammin ja vielä enemmän suoraan osaamiseen kehittymisen jälkiin ja pedagogiseen prosessiin paneutuen.

Tavoitteenani on ollut lähestyä aihetta oppimisanalytiikkaa tarkentavan osaamisanalytiikan näkökulmasta. Viestini on se, että oppimisanalytiikan, koulutuksen tiedonlouhinnan, akateemisen analytiikan ja korkeakoulututkimuksen aloilla tulisi jatkossa keskittyä entistä enemmän korkeakoulussa tuotetun osaamisen analysointiin. Korkeakoulutuksessa odotukset datan nykyistä paremmalle hyödyntämiselle tähän tarkoitukseen ovat korkealla. Korkeista odotuksista huolimatta jalat pysyvät maan pinnalla muistettaessa Rayónin ja muiden [2014, 297] toteamus: ”paraskin evidenssi oppimisesta vaatii tulkintaa – analyysiä, joka ottaa huomioon oppimisen kontekstin.”

Viiteluettelo

John P. Campbell and Diana G. Oblinger. 2007. Academic analytics. Educause. http://net.educause.edu/ir/library/pdf/PUB6101.pdf. Accessed 2.5.2016.

Miguel Á. Conde, Ángel Hernández-García, and Amílcar Oliveira. 2015. Endless horizons?: addressing current concerns about learning analytics. In: *TEEM '15: Proceedings of the 3rd International Conference on Technological Ecosystems for Enhancing Multiculturality,* 259-262.

Hendrik Drachsler, Tore Hoel, Maren Scheffel, Gábor Kismihók, Alan Berg, Rebecca Ferguson, Weiqin Chen, Adam Cooper, and Jocelyn Manderveld. 2015. Ethical and privacy issues in the application of learning analytics. In: *LAK '15: Proceedings of the Fifth International Conference on Learning Analytics and Knowledge*, 390-391.

Enrique García, Cristóbal Romero, Sebastián Ventura, Carlos de Castro, and Toon Calders. 2011. Association Rule Mining in Learning Management Systems. In: Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy, and Ryan Baker (eds.). *Handbook of Educational Data Mining*. CRC Press. 93-106.

Dragan Gašević, Shane Dawson, and George Siemens. 2015. Let’s not forget: Learning analytics are about learning. *TechTrends*, 59, 1, 64-71.

Dragan Gašević, Shane Dawson, Tim Rogers, and Danijela Gašević. 2016. Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success. *The Internet and Higher Education*, 28, 68-84.

Richard D. Howard, Gerald W. McLauglin, William E. Knight, and Associates (eds.). 2012. *Handbook of Institutional Research*. Jossey-Bass.

Frank Höppner. 2010. Association Rules. In: Oded Maimon and Lior Rokach (eds.). 2010. *Data Mining and Knowledge Discovery Handbook*. Second edition. Springer. 299-319.

Oded Maimon and Lior Rokach (eds.). 2010. *Data Mining and Knowledge Discovery Handbook*. Second edition. Springer.

Adam D. Moore. 2005. *Information Ethics: Privacy, Property, and Power.* University of Washington Press.

Ross J. Quinlan. 1992. Learning with Continuous Classes. In: *5th Australian Joint Conference on Artificial Intelligence,* 343-348.

Alex Rayón, Mariluz Guenaga, and Asier Núñez. 2014. Supporting Competency-Assessment through a Learning Analytics Approach Using Enriched Rubrics. In: *TEEM '14: Proceedings of the 2nd International Conference on Technological Ecosystems for Enhancing Multiculturality,* 291-298.

Alex Rayón, Mariluz Guenaga, and Jon Kepa Logarte. 2015. Standardized Enriched Rubrics to support competency-assessment through the SCALA methodology and dashboard. In: *Proc. of the IEEE Global Engineering Education Conference (EDUCON)*, 340-347.

Donald J. Reichard. 2012. The History of Institutional Research. In: Richard D. Howard, Gerald W. McLauglin, William E. Knight, and Associates (eds.). *Handbook of Institutional Research*. Jossey-Bass, 3-21.

Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy, and Ryan Baker (eds.). 2010. *Handbook of Educational Data Mining*. CRC Press.

Teresa Scantamburlo. 2016. Machine learning in decisional process: a philosophical perspective. *ACM SIGCAS Computers and Society,* 45, 3, 218-224.

George Siemens and Ryan Baker. 2012. Learning analytics and educational data mining: towards communication and collaboration. In: *LAK '12: Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, 252-254.

Sharon Slade and Paul Prinsloo. 2013. Learning Analytics: Ethical Issues and Dilemmas. *American Behavioral Scientist*, 57a, 10, 1510-1529.

Christina M. Steiner, Michael D. Kickmeier-Rust, and Dietrich Albert. 2016. LEA in Private: A Privacy and Data Protection Framework for a Learning Analytics Toolbox. *Journal of Learning Analytics*, 3(1), 66-90.

Alfredo Vellido, Félix Castro, and Àngela Nebot. 2011. Clustering Educational Data. In: Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy, and Ryan Baker (eds.). *Handbook of Educational Data Mining*. CRC Press. 75-92.

Digitaalisten oppimispelien kehitys ja hyödyntäminen kouluopetuksessa

Jarkko Pakalén

Tiivistelmä.
Digitaaliset oppimispelit tarjoavat uuden tavan oppia kouluopetuksen tueksi. Ne pyrkivät herättämään oppijassa kiinnostuksen pelaamisen kautta ja tuomaan opetettavan asian huomaamattomasti pelaajan saataville. Ongelmia oppimispelien kehityksessä on kuitenkin jonkin verran, koska ne eroavat hyötypeleinä melko paljon perinteisistä viihdepeleistä. Niiden käyttö on kuitenkin perusteltua, koska nykyinen diginatiivien sukupolvi on kasvanut pienestä asti sisään pelien ympäröimään maailmaan, ja heille pelien käyttäminen opetuksessa on enemmän luonnollista kuin epäluonnollista. Ongelmia oppimispelien käytölle luovat esimerkiksi vakiintuneet tavat opettaa ja se, että oppimispelit ovat vielä ominaisuuksiltaan kehityskaaren alkupäässä.

1. Johdanto

Pelaaminen on lisääntynyt huomattavasti viimeisen kahdenkymmenen vuoden aikana. Termi pitää sisällään nykyään paljon enemmän kuin se piti vielä jonkin aikaa sitten, jolloin ehkä lähin mieleen tuleva asia pelaamisesta puhuttaessa oli teini-ikäisten poikien istuminen sisällä konsolinsa tai tietokoneensa äärellä pelaten sisällöltään yllätyksetöntä verkkoräiskintäpeliä. Nykyään tuon mielikuvan tilalla on melko helposti ajatus kenestä tahansa missä tahansa pelaamassa mobiilipeliä tai jotain muuta peliä esimerkiksi bussipysäkillä. [Egenfeldt-Nielsen *et al.* 2013]

Samaan aikaan pelaamisen ja pelikehityksen ympärillä pyörivät summat ovat kasvaneet huomattavasti ja yhä useammin pelejä kehitetään mielessä myös muut asiat kuin pelkkä viihdearvo. Näitä hyötypeleiksi kutsuttavia pelejä pyritään luomaan, jotta yksilöt ja organisaatiot voisivat niiden avulla tarjota mahdollisuuden oppia joitain hyödyllisiä taitoja, joko työelämää tai koulua varten. [Egenfeldt-Nielsen *et al.* 2013]

Yksi melko ilmeinen hyötypelien osa-alue on oppimispelit (tai opetuspelit), joita kehitetään nimenomaan oppimistilanteita ja kouluja varten. Näissä peleissä on hyvin tyypillistä, että oppimistilanne on tavallaan naamioitu viihdyttävämmäksi pelimäiseksi kokemukseksi grafiikoiden ja erilaisten pelimekaniikkojen avulla. [Egenfeldt-Nielsen *et al.* 2013]

Usein oppimispelit kehitetään täsmällisesti vastaamaan johonkin tiettyyn opetustarpeeseen, kuten vaikkapa matematiikkaan tai kieliin. Toinen mahdollinen tapa hyödyntää pelejä oppimispeleinä on ottaa käyttöön jokin viihdetarkoitukseen tarkoitettu peli, joka sisältää opetuskäyttöön jo valmiiksi hyödyllistä sisältöä. Sisältö voi olla esimerkiksi historiallista tai tarjota jollain tavalla hyödyllisen simulaatiokokemuksen jostain aiheesta. [Egenfeldt-Nielsen *et al.* 2013] Simulaatiopelejä voivat olla esimerkiksi erilaiset lentosimulaattorit tai yhteiskunnan rakentamiseen tähtäävät simulaatiot [Mäyrä *et al.* 2010].

Tässä kirjallisuuskatsauksessa tarkastellaan erilaisia teorioita oppimispelien ympärillä, sitä mitä niiden kehityksessä tulee ottaa huomioon ja miten niitä voi hyödyntää koulussa. Pohdin, miten pelimekaniikat ja opetuksen voi pelin avulla yhdistää toimivaksi kokonaisuudeksi. Seuraavassa luvussa käsitellään ensin termistöä ja sitä, mikä on ensisijaista oppimispelien onnistumisen kannalta. Tämän jälkeen tutustutaan arviointikriteeristöihin ja siihen, miten oppimispelejä voi objektiivisesti arvioida. Käsittelen myös niiden kehitykseen liittyviä teorioita ja sitä, miten oppimispelit voivat auttaa oppijaa oppimaan. Lopuksi luodaan vielä katsaus siihen, millaisia oppimispelejä on käytännössä tarjolla suomalaisille kouluille tukemaan opetusta.

Tutkielman tavoitteena on tutustua oppimispeleistä tehtyyn tutkimukseen ja sen avulla keskittyä eri osa-alueisiin oppimispelien kehityksessä. Tavoitteena on myös esitellä pari onnistuneesti kehitettyä oppimispeliä ja niiden hyödyntämistä kouluissa.

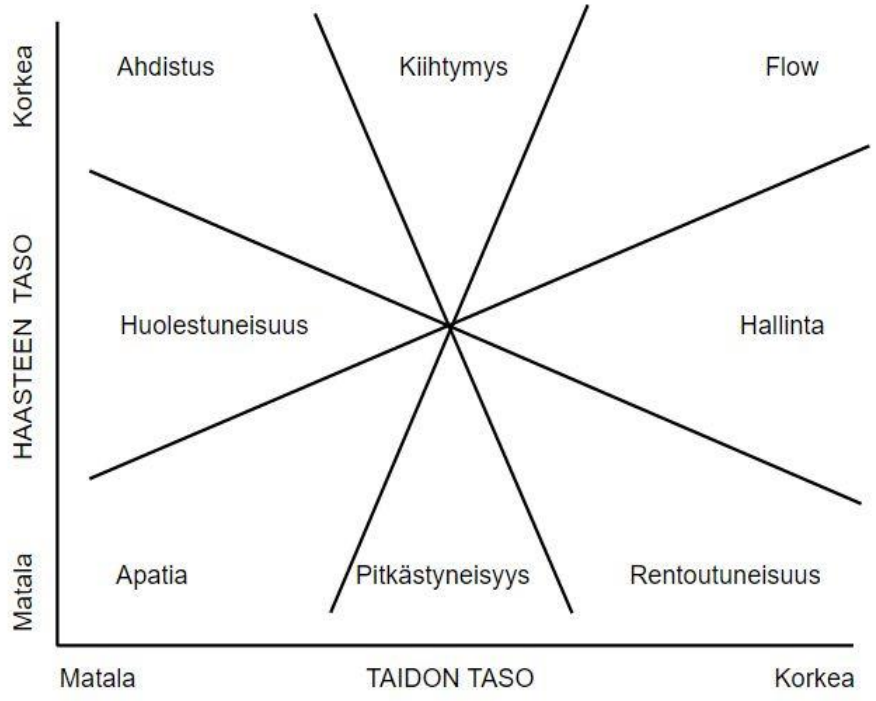
2. Flow ja immersio

Flow ja immersio ovat ominaisuuksiltaan melko lähellä toisiaan, mutta kuitenkin kaksi eri merkityksen omaavaa asiaa. Pelitutkimuksessa nämä kaksi käsitettä ovat keskeisiä, koska ne liittyvät kysymyksiin siitä, millainen on hyvä peli, miksi ihmiset pelaavat ja millä tavalla peli kannattaisi rakentaa, jotta se olisi viihdearvoltaan hyvä.

2.1 Flow-teoria

Flow-teoria on aihe, jota on tapana liittää moneen erilaiseen jollain tavalla ihmisen ajattelua sisältävään aiheeseen. Tämä johtunee siitä, että flow-tila on ihmiselle luontainen ja ehkä jopa tavoiteltava tila. Ihmisen ollessa sopivalla tavalla haastettu ja kun hänen taitonsa ovat riittävät ratkaistakseen hänelle asetettua haastetta, on ihmiselle luontaista kokea korkean keskittymisen tila. Tällöin ihminen on niin keskittynyt tekemäänsä asiaan, ettei kykene havainnoimaan ympäröivää maailmaa ja ajantaju katoaa. [Csikszentmihalyi 1990; 1997]

Flown perustana on ihmisen sisäinen motivaatio tehdä jotain itselle sopivalla tavalla mieluisaa asiaa. Flow-tilaan pääsy edellyttää kuitenkin samalla sitä, että sekä esitetyn haasteen, että ihmisen taitotaso haasteen ratkaisemiseksi ovat molemmat riittävän korkeat ja kohtaavat tasapuolisesti. Tällöin ihminen on erittäin keskittynyt tekemäänsä asiaan, koska hän ei ahdistu liian vaikeasta tehtävästä tai muutu liian rentoutuneeksi liian helposta tehtävästä. Flown vastakohtia ovat Csikszentmihalyin mukaan apatia, ahdistus ja rentoutuneisuus, kun taas lähinnä flow-tilaa ovat kiihtymys ja hallinta, kuten kuvasta 1 voidaan havaita. Hallinnan tunne syntyy ihmisen taitotason ollessa korkea verrattaessa keskinkertaiseen haastavuuden tasoon. Samalla tavalla kiihtymystila syntyy vastakohtaisesti haasteen tason ollessa korkea ja ihmisen taitotason ollessa vielä hieman vajaa verrattuna haasteeseen. [Csikszentmihalyi 1990; 1997]



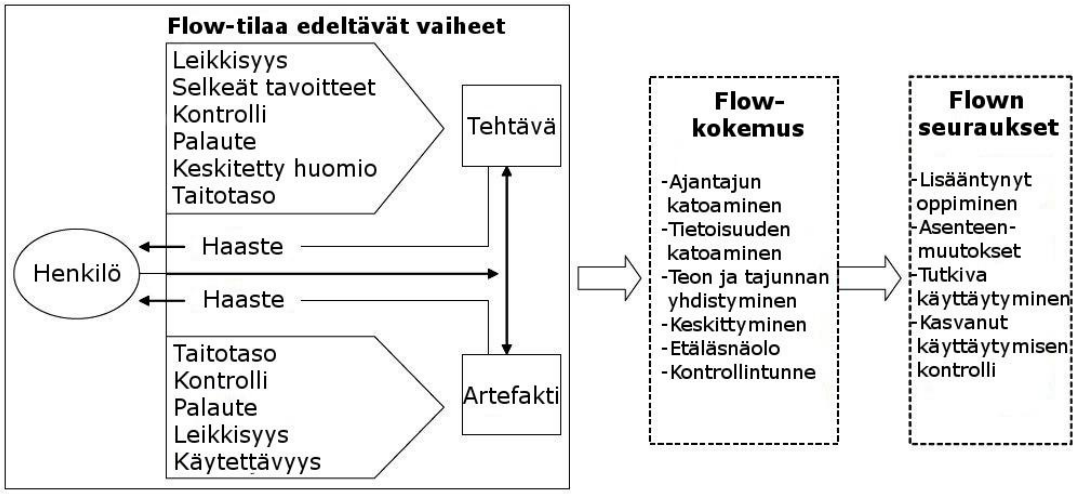
Kuva 1. Csikszentmihalyin [1997] Flow-malli.

2.2 Flown osat ja kokemuksen syntyminen

Flown on esitetty ja osoitettu rakentuvan kolmesta eri osa-alueesta. Nämä osa-alueet ovat edeltävät vaiheet (flow antecedents), kokemus (flow experience) ja seuraukset (flow consequences). Nimensä mukaisesti varhaisemmat vaiheet luovat edellytykset flow-kokemukselle, itse kokemus tarkoittaa flow-tilassa olemista ja seuraukset sisältävät flow-kokemuksen jälkeiset hyödyt ja haitat. [Ghani 1991; Trevino and Webster 1992; Ghani and Deshpande 1994; Chen 2000]

PAT-mallissa (Person-Artefact-Task –malli) esitetään flow-tilaa edeltävien vaiheiden rakentuvan erityisesti tietokoneympäristössä kolmesta eri palasesta. Nämä kolme palaa ovat henkilö, artefakti ja tehtävä. Tietokoneympäristössä tehtävät toiminnot saattavat vaatia ja useimmiten vaativatkin artefaktin osalta hyviä taitoja, jotta henkilö voi saavuttaa tekemänsä tehtävän osalta flow-tilan. Artefaktina voi toimia esimerkiksi taulukkolaskentaohjelma, jonka avulla henkilö laskee tietokoneympäristössä matkalaskuja. Tehtävä on täysin erilainen henkilölle, joka hallitsee taulukkolaskentaohjelman käytön kuin henkilölle, joka on tekemisissä sellaisen kanssa ensimmäistä kertaa. Flow-tilaan pääsy edellyttää henkilöltä tässä tapauksessa myös käyttämänsä artefaktin hyvää osaamista. [Finneran and Zhang 2003]

Kiilin [2005] luomassa mallissa, joka on esitetty kuvassa 2, kootaan yhteen flown osa-alueiden vaatimukset ja vaikutukset. Kiili pitää varsinkin oppimispelien kehityksessä tärkeänä, että siinä otetaan huomioon kaikki PAT-mallissa esitetyt osat. Niin henkilö, tehtävä kuin artefaktikin tuovat osuutensa oppimiskokemuksen syntymiseen oppimispeleissä. Henkilön ja artefaktin, eli oppijan ja pelin välisen vuorovaikutuksen tulisi olla mahdollisimman jouhevaa, jotta itse tehtävän suorittamiseen kohdennettavat resurssit olisivat mahdollisimman suuret. Näin tarjotaan oppijalle mahdollisuus saavuttaa ensisijaisesti hyvä oppimistulos. [Kiili 2005]



Kuva 2. Kiilin [2005] flow-tilaa edeltävien vaiheiden malli.

Oppijan ja artefaktin huonon vuorovaikutuksen on osoitettu heikentävän flow-tilaan pääsyä oppimispeliä pelattaessa ja näin hankaloittavan pelimaailmasta syntyvän immersion ja oppimisen tuloksia. Lukio(high school)-oppilailla suoritettu kokeilu Frequency 1550 -oppimispelin parissa osoitti, että mikäli oppilaat eivät ole tutustuneet pelissä käytettäviin välineisiin riittävällä tavalla, estää

se flown syntymistä. Pelissä oli tarkoituksena suorittaa tehtäviä keskiaikaisessa Amsterdamissa ja sitä kautta oppia aiheeseen liittyviä asioita. Oppilaat suorittivat näitä tehtäviä ryhmissä. Teknologian ajoittainen toimimattomuus ja muut ongelmat aiheuttivat kuitenkin hankaluuksia pelin kanssa ja estivät näin flown syntymisen peliä pelattaessa. Flowta havaittiin kuitenkin silloin, kun teknisiä ongelmia ei ilmennyt. [Admiraal *et al.* 2011]

2.3 Immersio digitaalisissa peleissä

Hyvin läheinen teoria flown kanssa on immersio, jota on viime aikoina pidetty varsinkin pelitutkimuksessa erittäin tärkeänä asiana. Immersiosta puhuttaessa tulee nopeasti mieleen ajatus uppoutumisesta johonkin asiaan. Mitä uppoutuneempi pelaaja on pelimaailmaan, sitä immersiivisempi tämän kokemus on.

McMahanin [2003] mukaan immersion vahvuutta lisää luonnollisesti visuaalisen ja auditiivisen kokemuksen vahvuus. Tällöin esimerkiksi suurempi näyttö ja parempi äänentoisto tarjoavat paremmat lähtökohdat immersion syntymiselle.

Tätä tärkeämpää on kuitenkin pelin sisältö ja siihen liittyvä vuorovaikutus. Immersion syntymiseksi täytyy pelin toteuttaa kolme asiaa. Pelin sisällön täytyy vastata pelaajan odotuksia pelistä, pelaajan valintojen ja vuorovaikutuksen pitää näkyä pelissä ja pelin täytyy säilyttää looginen jatkumo siitä, millainen se on ollut alusta asti. [McMahan 2003] Tiivistetysti immersiota voitaneen kuvata pelikokemuksen osana, joka määrittelee, kuinka uppoutunut henkilö on sillä hetkellä tekemäänsä asiaan [Ermi and Mäyrä 2005].

Immersiota ja flowta ei pidä kuitenkaan sekoittaa keskenään, sillä vaikka ne ovat melko samankaltaiset tilat, ne käsittelevät eri asioita. Flow-tilassa ihminen keskittyy lähinnä yhteen (tai muutamaan) melko tarkkaan tiedossa olevaan tavoitteeseen. Immersiossa kyse on enemmänkin siitä, että ihminen tuntee olevansa osa kokemusta. Tämä voi tapahtua niin virtuaalisesti kuin fyysisestikin. Kiilin mukaan flow-teoria on oppimispelinkehityksen kannalta lopulta mielenkiintoisempi aihe, koska se vastaa aiheen sisällä useampaan kysymykseen, mutta immersiota ei sovi myöskään unohtaa. Immersiota tavoiteltaessa, tulisi nimenomaan oppimispeleissä kuitenkin ottaa myös huomioon ihmisen muistin kognitiiviset rajoitteet. Esimerkiksi liian paljon materiaalia kerralla sisältävät peliympäristöt ovat haitallisia oppimistuloksen kannalta, koska oppija joutuu keskittymään liian moneen asiaan samalla kerralla. Tämä johtuu siitä, että ihmisellä on rajallinen kyky käsitellä tietoa kerralla. [Kiili *et al.* 2012]

Oppimispeleissä tärkeää on luonnollisesti saada oppija keskittymään pääasiassa opeteltavaan asiaan, jolloin liiallinen immersiivisyys voi suunnata oppi-

jan huomiota pelimaailmassa muihin epärelevantteihin asioihin. Näkisin kuitenkin, että varsinkin simulaatioissa immersiivisyyttä pitää olla riittävästi uskottavuuden saavuttamiseksi.

3. Pelin pelattavuuden ja käytettävyyden arviointi

Aiemmassa luvussa keskityin kertomaan siitä, miten ja miksi pelaaja kokee pelaamansa pelin miellyttäväksi. Pelaajalle on tärkeää kokea sekä flow että immersio, jotta pelaaja voi kunnolla uppoutua pelikokemukseen. Tämän onnistuminen ei kuitenkaan ole itsestäänselvyys, vaan vaatii jonkinlaista laaduntarkkailua ja varmistusta koko pelinkehityskaaren ajan. Helpottaakseen tätä prosessia voi peliä kehittävä taho turvautua valmiiksi laadittuihin heuristiikkoihin.

Heuristisessa arvioinnissa joukko käytettävyysasiantuntijoita käy vaihe vaiheelta läpi heuristisen arvioinnin kohteena olevan ohjelmiston. Heuristinen arviointi suoritetaan yleensä valmiiksi laadittujen tarkistuslistojen avulla (liite 1 ja liite 2) ja vertaamalla tapauskohtaisesti käyttäjän tarpeita ohjelmistoon. Usein heuristisessa arvioinnissa löydetyt puutteet vielä järjestetään vakavuuden mukaan suurimmasta pienempään. [Maguire 2001]

Pelattavuus ja käytettävyys etenevät melko pitkälle käsi kädessä pelisuunnittelussa, mutta ovat kuitenkin samalla kaksi eri asiaa. Pelattavuutta on yritetty määritellä muutamallakin erilaisella tavalla, mutta yleisesti määritelmissä korostuvat pelaamisen kokemus ja toiminnallisuus. Esimerkiksi Paavilainen ja Saarenpää [2009] määrittelevät pelattavuuden siten, että se vastaa pelien mielekkyyteen ja toiminnallisuuteen, mutta samalla myös käytettävyyteen. Molemmat ovat oikeastaan yhtä tärkeitä asioita, jotta pelikokemus olisi pelaajalle mahdollisimman antoisa. Yleisesti ottaen käytettävyyden avulla on totuttu analysoimaan enemmänkin tehtäväorientoituneita ohjelmistoja, kun taas pelattavuus vie arvioinnin seuraavalle tasolle tuoden mukaan arviointiin myös immersion, haasteen ja viihdearvon [Desurvire and Wiberg 2009].

Erilaisia pelattavuus- ja käytettävyysheuristiikkoja peleille on olemassa muutamia ja tässä luvussa esitellään kaksi sellaista. Heuristiikat ovat yleisesti hyväksytty tapa arvioida jonkin ohjelmiston hyödyllisyyttä ja toimivuutta asiaan, jonka yhteydessä sitä on tarkoitettu käytettäväksi. Ilman helppokäyttöisiä heuristiikkoja on huomattavasti vaikeampaa päästä onnistuneeseen lopputulokseen, koska ohjelmistokehityksessä (ja pelikehityksessä) on tärkeää olla tietoinen sen hetkisestä tuloksesta ja kuinka hyvin työ on siihen mennessä onnistunut.

Heuristiikkojen avulla voidaan tilannetta pyrkiä korjaamaan kehitystyön aikana tai sen jälkeen. Oppimispeleille ei ole olemassa vielä ainakaan vakiintu-

neita heuristiikkoja, mutta videopeliheuristiikkoja voinee yleisesti ottaen hyödyntää myös oppimispelin suunnittelussa, kunhan muistaa oppimispelin ensisijaisen tarkoituksen olla opettavainen.

3.1 PLAY-heuristiikka (Game Usability Heuristics)

Alun perin vuosituhannen alussa kehitetty HEP-heuristiikka, eli *Heuristic to Evaluate Playability* (suom. *heuristiikat pelien pelattavuuden arviointiin*) tarjosi pelikehittäjille ensimmäiset modernit pelattavuusheuristiikat, joita pystyi käyttämään pelikehityksen aikana. HEP-heuristiikoissa keskityttiin neljän eri pelien osa-alueen arviointiin, jotka olivat *pelattavuus*, *tarina*, *pelimekaniikat* ja *käytettävyys*. [Desurvire *et al.* 2004]

HEP-heuristiikkoja laajennettiin PLAY-heuristiikoilla, eli *Game Usability Heuristics* (suom. *pelien käytettävyysheuristiikat*). Nämä tarjoavat pelikehittäjille työkalut, joilla nämä voivat HEP-heuristiikkojen tapaan jo ennen kehitystä ja kehityksen aikana keskittyä pelattavuuden ja muiden kategorioiden kannalta olennaisiin pelin osa-alueisiin. Heuristiikat ovat hyvä työkalu pelinkehityksen tukena ja mahdollistavat jopa uusien ideoiden syntymisen pelimekaniikkoja rakennettaessa. [Desurvire and Wiberg 2009]

Desurvire ja Wiberg [2009] kertovat, että HEP-heuristiikkoja koottaessa tuli esille useita erillisiä asioita, jotka vaikuttavat siihen, onko peli hyvä vai huono. Yksi tärkeistä ja huomionarvoisista asioista on se, että pelaajat odottavat peliltä jonkinlaista vaikeustasoa. Peliä ei saisi kuitenkaan tahallisesti vaikeuttaa käytettävyyden kautta tai pakottamalla pelaajaa muistamaan joitain asioita, vaan lähinnä lisäämällä haasteen ja strategian vaatimusta.

Desurvire ja Wiberg [2009] toteavat, että pelaajat toivovat myös pelaamisen aikana jonkinasteista pelin vaikeustason nostoa. Tässä tärkeänä asiana on pelaajan taitojen kehittymisen lisäksi myös niiden kehityksen oikeantasoinen tempo. Pelikokemus pysyy tuoreena ja kestävänä, mikäli pelaajan taitotaso kehittyy sopivalla tahdilla pelaajan pelikokemukseen ja tehtäviin suhteutettuna.

Pelin juoni näyttelee suurta osaa pelikokemuksen ja immersion syntymisessä. Pelaajat kokevat juonen ja kunnollisen perustelun kautta annettujen tehtävien tarjoavan enemmän motivaatiota niiden suorittamiseen kuin yksittäiset ja ilman perustelua tarjotut tehtävät. Hyvän käyttäjäkokemuksen syntymisen takaamiseksi, tämä tuntui olevan erittäin tärkeä kategoria. [Desurvire and Wiberg 2009]

Lopulliset PLAY-heuristiikat on jaoteltu kolmeen eri kategoriaan ja ala-kategorioihin, jotka sellaisenaan keskittyvät eri osa-alueisiin [Desurvire and Wiberg 2009]. Ne ovat kokonaisuudessaan tutkielman liitteenä (Liite 1).

PLAY-heuristiikat tarjoavat työkalut yleisesti pelien pelattavuuden ja sisällön arviointiin, ja ne onkin kehitetty useiden peliteollisuuden yritysten kanssa keskittymällä tutkimuksessa kolmeen eri peligenreen, joita ovat reaaliaikaiset strategiapelit, toimintaseikkailupelit ja ensimmäisen persoonan ammuntapelit. [Desurvire and Wiberg 2009]

PLAY-heuristiikkoja voinee hyödyntää myös oppimispelien kehityksessä, arvioitaessa pelimekaniikkoja ja yleistä pelattavuutta. Esimerkiksi oppimispeleissä näkisin juonen ja tarinan tärkeäksi osa-alueeksi, koska se todennäköisesti motivoisi pelaajaa jatkamaan peliä. Yksi mahdollisuus oppimispelissä voisi olla, että esimerkiksi suorittamalla tehtäviä pelaaja saisi juonta etenemään. Sopivasti epävarmaan tilanteeseen jätetty juoni loisi pelaajalle sisäistä motivaatiota jatkaa pelin pelaamista, jotta juonen seuraava osa avautuisi. Myös liitteessä 1 oleva heuristiikka *3(i) tarina ja sen immersio* keskittyy kyseiseen asiaan.

PLAY-heuristiikat eivät varsinaisesti kuitenkaan vastaa opetukselliseen sisältöön, vaan lähinnä tarjoavat keinot arvioida videopelien pelattavuutta ja käytettävyyttä. PLAY-heuristiikoissa ei ole myöskään erikseen eritelty moninpelaamiselle heuristiikkoja. Näistä puhutaan tarkemmin seuraavassa kohdassa.

3.2 Mobiilipeliheuristiikat

Toinen hyvä peliheuristiikkakokoelma on Koiviston ja Korhosen [2006; 2007] ensisijaisesti mobiilipeleille kehittämät heuristiikat. Mobiilipeliheuristiikat ottavat huomioon myös moninpelit ja niihin liittyvät pelattavuusongelmat. Heuristiikoissa on keskitytty neljään erilliseen teemaan, jotka ovat *"pelin käytettävyys"*, *"pelattavuus"* *"liikuteltavuus"* ja *"moninpeli"*. Näistä käytettävyys, pelattavuus ja liikuteltavuus olivat mobiilipeliheuristiikkojen kehityksen ensimmäisen iteraation tulokset, jonka jälkeen heuristiikkoihin lisättiin vielä moninpeliheuristiikat. Ensisijaisesti kehittäjät ovat tarkoittaneet heuristiikat käytettäväksi valmiiden pelien pelattavuuden ja käytettävyyden arviointiin ja puolivalmiiden pelien pelikehityksen tueksi. [Koivisto and Korhonen 2006; 2007]

Koiviston ja Korhosen [2006; 2007] mukaan pelin olemus rakentuu neljästä erillisestä kerroksesta, jotka pelaaja kohtaa pelatessaan peliä. Pelin saavutettavuus määräytyy sen mukaisesti, ovatko eri kerrosten kuvaamat osat vakaita vaiko epävakaita. Mikäli ulommat kerrokset ovat epävakaita, saattaa sisempiin kerroksiin siirtyminen olla vaikeaa, ellei jopa mahdotonta. Uloin kerros on ensisijainen väline vuorovaikutukselle, eli itse laitteisto, jolla peliä pelataan. Toiseksi uloin kerros kuvaa pelin ohjelmistoa. Toiseksi sisin kerros sisältää pelin käytettävyyden ja liikuteltavuuden. Sisin kerros on pelin yleinen pelattavuus. Heuristiikkoja käytettäessä on huomioitava, että niillä voidaan ensisijaisesti vaikuttaa

vain kahteen sisempään kerrokseen, eli käytettävyyteen, liikuteltavuuteen ja pelattavuuteen. Tämän ei kuitenkaan pitäisi olla esteenä arvioinnin suorittamiselle, vaikka ulommat kerrokset olisivat vielä kehitysasteella tai muulla tavalla epävakaita. [Koivisto and Korhonen 2006; 2007]

Johonkin näiden kerrosten väliin voitaneen sijoittaa oppimiseen liittyvät pelielementit. Näkisin pelien avulla tapahtuvaan oppimiseen liittyvän olennaisesti niin pelattavuuden kuin myös itse laitteiston. Ilman näiden vuorovaikutusta jää oppimispelin merkitys vajaaksi, koska oppimispelin hauskuus vaatii pelielementtien hyvää toteutusta ja vastaavasti helppokäyttöisyys laitteiston ja ohjelmiston onnistunutta toteutusta. Kiili [2005] on havainnut tutkimuksessaan samaa.

Mobiilipeliheuristiikkojen neljä kategoriaa jakautuvat useampaan erilliseen alakategoriaan, joista jokainen tarjoaa yhden heuristiikan pelin ominaisuuden tulkitsemiseksi. On huomioitava, että jotkin heuristiikat saattavat olla ristiriidassa keskenään tai jotkin peliominaisuudet saattavat toteuttaessaan yhden heuristiikan, rikkoa toisen heuristiikan. On kuitenkin mahdollista hyvien perusteluiden vuoksi rikkoa joitain heuristiikkoja, mikäli kehittäjät ovat sitä mieltä, että se on pelin kannalta oleellista tai tuo pelille joitain toivottavia ominaisuuksia. [Koivisto and Korhonen 2006; 2007]

Oppimispelien toteuttamisessa lienee syitä rikkoa muutamia heuristiikkoja. Esimerkiksi liitteessä 2 mainittu pelin käytettävyyteen liittyvä heuristiikka ”*GU11: Pelaajan ei tarvitse muistaa asioita turhaan*” saattaa olla oppimispelitarkoituksessa jopa jollain tasolla ristiriidassa opetuksellisen näkökulman kanssa. Oppijaa tulisi ehkä jopa rohkaista muistamaan asioita päästäkseen pelissä eteenpäin, jotta opetuksellinen näkökulma toteutuisi. Toki turhien ja opeteltavaan asiaan liittymättömien asioiden muistaminen on tässäkin tapauksessa aiheetonta.

Toisaalta joitain heuristiikkoja tulisi oppimispelien kehityksessä ehkä painottaa enemmän. Esimerkiksi liitteessä 2 mainittu pelin käytettävyyteen liittyvä ”*GU9: Peli antaa palautetta pelaajan toimista*” on melko keskeinen oppimispelien näkökulmasta. Oppija tarvitsee luonnollisesti palautetta oppimisprosessistaan, jotta tämä tietäisi, missä hän on kehittynyt ja missä ei. Uskoakseni palautetta ei kuitenkaan tulisi välttämättä antaa perinteisesti pelkästään pelaamisesta, vaan yhdistää palautteeseen myös tietoa opittavan asian etenemisestä. Näkisin myös pidemmällä aikavälillä tarjottavan palautteen olevan tärkeää, koska näin oppija voi seurata omaa kehitystään erilaisilla saamillaan palautteilla. Tämä luo pelaajalle samalla keinon pelillistää omaa oppimistaan. Pelillistäminen tar-

koittaa pelimäisten elementtien sisällyttämistä pelien ulkopuolisiin ympäristöihin, jolla voidaan tukea esimerkiksi käyttäjän sitoutumista ja osallistumista [Deterding *et al.* 2011; Huotari and Hamari 2012].

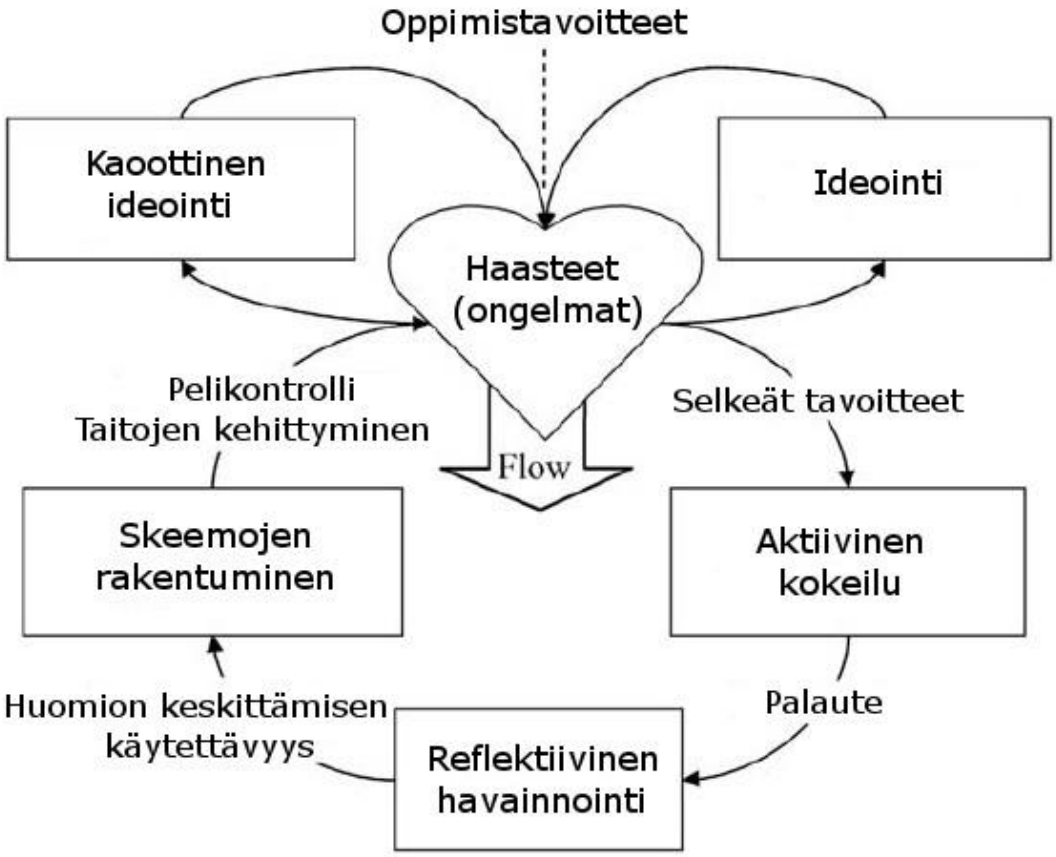
Toinen heuristiikka, liitteessä 2 mainittu liikuteltavuuteen liittyvä moninpeliheuristiikka "*MO3: Keskeytyksiin reagoidaan järkevällä tavalla*" on kouluympäristöön oppimispeliä sovellettaessa tärkeä, koska ulkoisen toimijan on oltava mahdollista pyytää väliaikaisesti keskeyttämään pelin pelaaminen. Tällöin pelin tulee tarjota järkevä tapa toteuttaa pyyntö ja samalla tapa jatkaa pelaamista keskeytyksen jälkeen. Tässä tapauksessa pelin tulisi ottaa myös huomioon, että mahdollinen flow-tila on keskeytyksen johdosta menetetty ja pelaajaa tulisi jollain tavalla tukea sen uudelleensaavuttamisessa.

4. Oppimispelien toteuttaminen

Objektiivinen virheenetsintä on luonnollisesti tärkeää kaikessa ohjelmistokehityksessä. Myös pelikehityksessä on tärkeää varmistua laadusta, pelattavuudesta ja viihdearvosta. Siihen erilaiset heuristiikat vastaavatkin erittäin hyvin. Oppimispeleihin liittyy kuitenkin myös vähintään yhtä kiinteänä osana opetus ja oppiminen. Tässä luvussa tarkastellaan mahdollisia lähestymistapoja oppimispelien opetuspuolen kehitykselle.

4.1 Kokemusperäinen pelaamismalli

Yksi teoreettinen malli oppimispelien sisällön kehitykselle on Kiilin [2005] esittelemä kokemusperäinen pelaamismalli (Experiential Gaming Model). Kuvassa 3 esitelty malli rakentuu kahden erillisen silmukan ympärille, joiden keskiössä toimii mallin sydän, jonka tarkoituksena esittää oppijalle uusia haasteita ja ongelmia ratkaistavaksi. Silmukoita kutsutaan ideointisilmukaksi ja kokemussilmukaksi. Sydäntä kutsutaan haastepankiksi. [Kiili 2005]



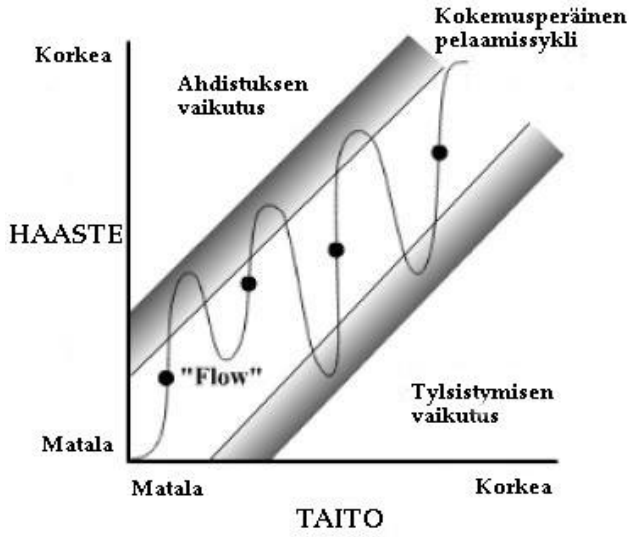
Kuva 3. Kiilin [2005] kokemusperäinen pelaamismalli.

Uuden haasteen saatuaan oppija alkaa ideointisilmukan kautta hakea ratkaisuja haasteeseen. Silmukka on mallissa jaettu kahtia: toinen osa silmukkaa vastaa oppijan kaoottisempaa ideointia (preinvative idea generation), jota voi verrata lasten leikkiin, ja toinen osa silmukkaa tavallista ideointia (idea generation) [Finke *et al.* 1992]. Kaoottisemmassa ideoinnin vaiheessa oppija saattaa saavuttaa erilaisia tuloksia verrattuna normaaliin ideointiprosessiin, koska oppija ei välttämättä ota huomioon tilanteen hänelle asettamia rajoituksia. Tällöin ideointi on innovatiivisempaa. Tämän ideointivaiheen jälkeen seuraa tavallisempi ideoinnin vaihe, jossa oppija vertaa ajatuksiaan tilanteen asettamiin rajoituksiin joko yksin tai ryhmässä. Ryhmässä suoritettu ideointi on tehokkaampi tapa suoriutua tästä vaiheesta. [Kiili 2005]

Kiili [2005] kertoo, että ideointisilmukan jälkeen pelaaja testaa käytännössä ideoitaan kokemussilmukassa. Tämän mahdollistamiseksi tulee pelin käytettävyyden olla hyvä ja samalla sen tulee tarjota pelaajalle käytettävää ja hyödyllistä palautetta tehtävänsuorituksen aikana [Kiili 2005]. Tätä voidaan tukea jo pelinkehitysvaiheessa, hyödyntämällä esimerkiksi aiemmin esiteltyjä PLAY- tai Mobiilipeliheuristiikkoja.

Pelin tulee rohkaista pelaajaa kohti tiedon reflektointia ja tiedon rakennusta ohjaamalla pelaajan huomiota kohti opeteltavia asioita. Oppijan pohtiessa saamaansa palautetta on hänen mahdollista löytää uusia ja parempia tapoja ratkaista esillä oleva asia ja oppia samalla uutta. Aiheeseen liittyy vahvasti myös pelimaailman jakaminen moninpeliin ja yksinpeliin. Vaikka moninpelissä on mahdollista hyödyntää kanssaoppijoita ja pohtia esillä olevia aiheita yhdessä, on yksinpeli tärkeä osa oppimista, sillä pohdiskelu ja tiedon rakentaminen tapahtuu lopulta itsenäisesti. [Kiili 2005]

Ideointisilmukka on lopulta teorian kannalta erittäin tärkeä, koska se puhdistaa oppijan kokemussilmukassa luomia ajatuksia ja sitä kautta tarjoaa hänelle aina uusia näkökulmia. Sydän on teorian keskiössä sen tärkein osa. Sydämen tehtävänä on tarjota oppijalle tämän taitotasoa vastaavia haasteita ja täten ylläpitää flow-tilaa mahdollisimman tehokkaasti. Ajatuksena on kasvattaa haasteen tasoa samalla, kun pelaaja kehittyy oppimistuloksissa, jotta haasteet eivät olisi liian vaikeita tai liian helppoja. [Kiili 2005]



Kuva 4. Kiilin [2005] kokemusperäinen pelaamissykli.

Saavuttaakseen optimaalisen oppimisen tason tulisi oppimispeliä pelaavan samalla säilyttää optimaalinen keskittymisen tila, jonka toteutuminen on esitetty kuvassa 4. Tämä voidaan saavuttaa, mikäli peli osaa mukautua oppijan toimintojen ja taitotason mukaan, tarjoamalla pelaajalle yksilöllisen vaikeustason pelattaessa ja opeteltaessa uutta asiaa. Pelaajan kokiessa ahdistuneisuutta tai tylsistymistä tulisi pelin kyetä korjaamaan tarjoamaansa haasteen tasoa, optimoidakseen tämä pelaajan taitotason kanssa. Tällä tavalla pelaajan taitotason kasvaessa myös pelin haasteen tason tulisi kasvaa asteittain. Pelin tulisi tehdä tämä kuitenkin niin huomaamattomasti, ettei pelaaja huomaa muutoksia. Mikäli pelaaja esimerkiksi

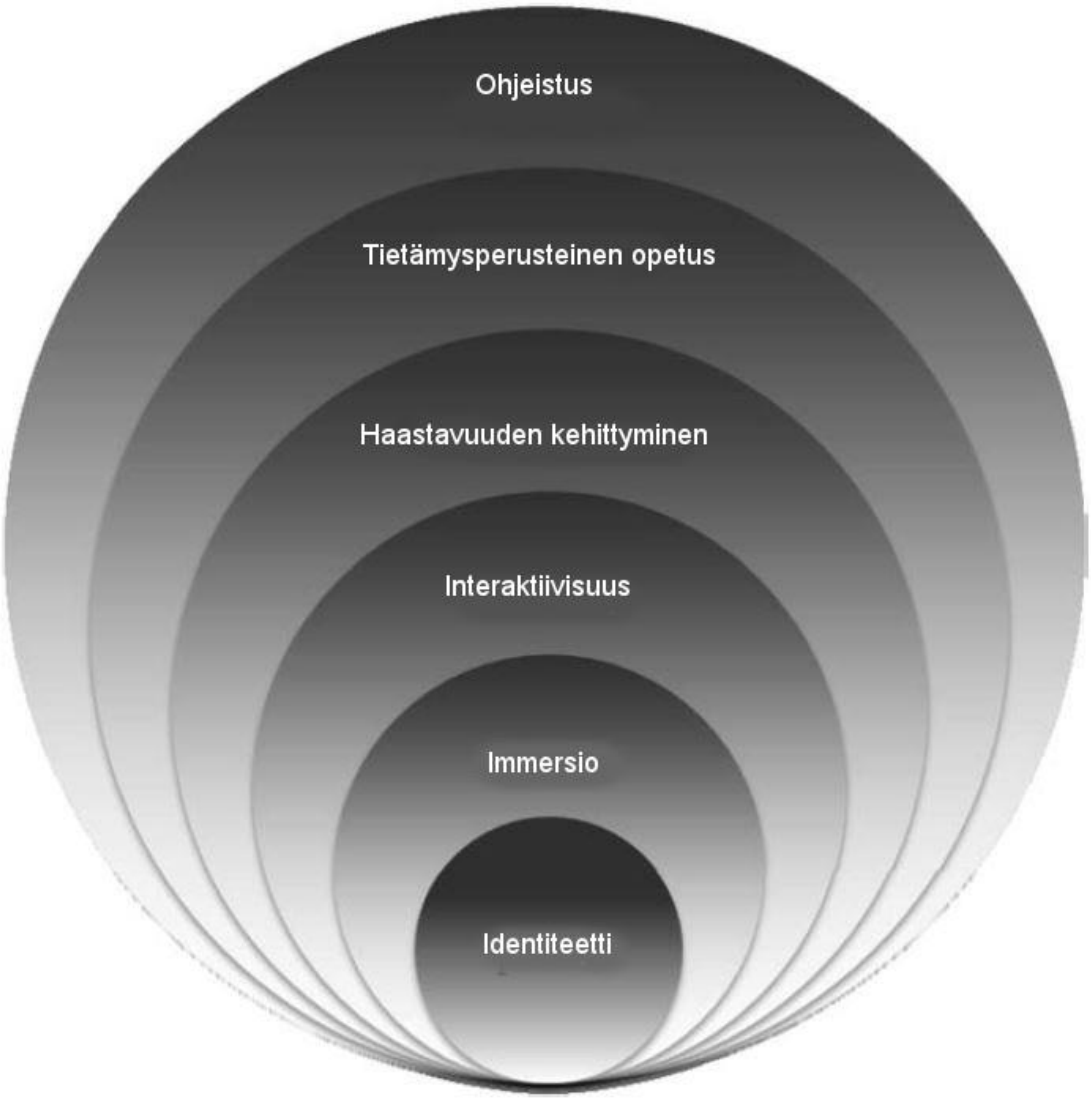
huomaa pelin helpottuvan tämän osaamattomuuden takia, on pelimoottori epäonnistunut mukautumisessa. [Kiili 2005]

Pelaaja ei saisi huomata hänen taitotason perusteella tehtäviä muutoksia, vaan pelin kulun tulisi olla yhtenäinen. Tämä liittyy vahvasti pelaajan kokemukseen immersiosta ja flowsta, koska kokemus rikkoutuu helposti, jos pelaaja alkaa havainnoida pelin ulkopuolisia asioita, kuten pelimoottoria tai muita pelikokemukseen kuulumattomia asioita. Mikäli peli kykenee tekemään edellä mainitun mukautumisen huomaamattomasti, kyetään pelaajan flow-tila ja immersio säilyttämään ja tätä kautta tukemaan pelaajan oppimista. Toisaalta voitaneen sanoa, että vielä huonompi vaihtoehto olisi, jos peli ei osaisi mukautua ollenkaan pelaajan taitotasoon. Näkisinkin, että edes näkyvästi toteutettu muutos vaikeustasossa on parempi kuin se, että muutosta ei tehtäisi ollenkaan.

4.2 Kehys oppimispelien rakentamiselle

Yksi lähestymismalli oppimispelin rakentamiseen on Annettan [2010] kuuden i:n kehys. Annettan mukaan hyvälle oppimispelille tärkein elementti on oppijan identiteetti pelin sisällä. Tämän päälle voidaan hänen mukaan rakentaa pelin muu sisältö. Tämä ei kuitenkaan tarkoita, että opetus tulisi jättää pelkästään oppimispelin vastuulle, vaan ulkopuolisen opettajan tulee tarkkailla suorituksia ja ohjeistaa pelin pelaajaa, samalla tarjoten tälle uusia haasteita.

Annetta [2010] esittelee kehyksessä kuusi kohtaa, jotka hänen näkemyksen mukaan ovat hyvän oppimispelin perusta. Hän tuo myös esille, että kehys toimii sipulimallin mukaisesti ja ennen kuin sisempi kohta on toteutettu ei ole järkevää keskittyä ulompaan kohtaan. Kuusi kohtaa, jotka on esitetty kuvassa 5, ovat järjestyksessä sisältä ulos: identiteetti (identity), immersio (immersion), interaktiivisuus (interactivity), haastavuuden kehittyminen (increasing complexity), tietämysperäinen opetus (informed teaching) ja ohjeistus (instructional).



Kuva 5, Annettan [2010] kuuden i:n kehys.

Annettan [2010] mukaan tärkein oppimispelin kehityksessä muistettava asia on peliä pelaavan oppijan *identiteetti*. Oppijalle on tärkeää tarjota pelaamisen ajaksi hahmo, johon hän voi samaistua. Ilman tätä pelaaja ei välttämättä koe yhtä voimakkaasti uppoutumista ja samaistumista peliin kuin tämä kokisi omakseen kokemallaan hahmolla.

Annetta [2010] kertoo, että identiteetin luomisen jälkeen voidaan alkaa pohtia pelin ja pelaajan *immersiota*. Identiteetti liittyy hänen mukaansa olennaisesti myös immersion syntymiseen pelissä. Omalla hahmolla pelatessa pelaajan on helpompi uppoutua peliin kuin jos hän pelaisi yleisesti valmiiksi annetulla hahmolla. Tämä herättää pelaajassa sisäistä motivaatiota suorittaa pelin tarjoamia tehtäviä, koska pelaaja ottaa nämä tehtävät vakavammin ja kokee ne enemmän omakseen. Kun pelaajat ovat sopivalla tavalla haastettuja, heiltä löytyy motivaatiota ja he ovat uppoutuneena peliin, on pelaajien mahdollista saavuttaa flow-

tila. Flow tarjoaa pelaajalle ja oppijalle täydellisen keskittymisen mahdollisuuden ja kokemuksen olla niin sanotusti täydellisen uppoutuneena tekemäänsä asiaan.

Myös vuorovaikutus on oppimisen kannalta tärkeää ja tämän vuoksi oppimispelien pitäisi olla jollain tasolla *interaktiivisia*. Annettan [2010] mukaan flow-tilaan pääseminen voi kuitenkin olla haastavampaa moninpeleissä kuin yksinpeleissä, koska niiden vaatimat ylimääräiset auditiiviset ärsykkeet saattavat haitata täydellistä keskittymistä visuaaliseen ärsykkeeseen. Hän kuitenkin korostaa, ettei se ole myöskään mahdotonta. On myös huomioitava, ettei moninpeli ole mitenkään pakollinen muoto oppimispelille, koska pelaajat tuntuvat reagoivan sosiaalisen toiminnan puolesta samalla tavalla niin ihmisen kuin myös tietokoneen ohjaamiin hamoihin. Interaktiivisuutta löytyy tällöin samalla tavalla myös yksinpelistä kuin moninpelistäkin. [Annetta 2010]

Annettan [2010] mukaan yksi vaikeimmista, ja samalla tärkeimmistä osa-alueista hyvän oppimispelin kehityksessä on sopivan vaikeustason rakentaminen peliin, eli *haastavuuden kehittyminen*. Hän kertoo, että peleille on luontaista, että ne muuttuvat sitä haasteellisemmiksi, mitä pidemmälle pelaaja pelissä etenee. Edelliseen liittyen, samalla tavalla voidaan olettaa oppimista tapahtuvan mitä pidemmälle oppija etenee oppimispelissä. Kuten jo edellisessä luvussa pohditaan, vaikeaksi tämän osa-alueen kehittämisen tekee se, että pelin täytyisi tasapainotella sopivassa ahdistuneisuuden ja tylsistymisen välimaastossa. Kiili [2005] toteaa, että pelin täytyy mukauttaa taitotasoaan sitä mukaan kuin pelaajan osaamisen taso vaihtelee.

Pelaaja jaksaa haastaa itseään pidemmälle pelissä, jos taso pysyy sopivana eikä muutu jossain kohdin liian vaikeaksi tai liian helpoksi. Yksi keino saavuttaa tämä voisi olla monien nykyaikaisten roolipelien hiekkalaatikkomainen toteutus, joissa pelaaja suorittaa erillisiä tehtäviä senhetkisen mielenkiintonsa mukaan. Uusia kehittyneempiä haasteita voisi avautua sitä mukaa kuin peli arvioisi jonkin edellisen asian tulleen opituksi. Peli voisi näin vaikeutua asteittain sitä mukaa, kun se huomaisi pelaajan taitojen kehittyvän. [Annetta 2010]

Ongelmia edelliselle toteutukselle saattaisi kuitenkin asettaa tehtävien järkevä tarjoamisjärjestys. Oppija saattaisi olla niin kiinnostunut yhdestä tehtäväpolusta, että jättäisi muut tehtävät kokonaan suorittamatta. Tällaiselle toimintamallille tulisi asettaa rajoituksia esimerkiksi suoritettavien tehtävien määrällä, tai joillain muilla keinoilla.

Jotta peli voisi itsenäisesti kehittyä yksilön taitotason mukaan, täytyy sen jollain tavalla kerätä informaatiota pelaajan tottumuksista, pelityylistä ja kehittymisestä. Tämä mahdollistaisi *tietämysperusteisen opetuksen*, mutta vaatii samalla kehittyneitä pelinsisäisiä algoritmeja, koska olisi epärealistista olettaa, että opet-

tajalla tai muilla toimijoilla olisi aikaa tai resursseja tarkkailla jokaista yksilöä ulkoisesti erikseen. Tällainen toiminta mahdollistaisi myös muunlaisia lähestymistapoja opetukseen, kuten yksilöllisempää opetusta ymmärrettäessä opiskelijan oppimistapoja pelin keräämään tietoon perustuen. [Annetta 2010]

Annettan [2010] mukaan oppimispelien kohdalla voidaan oppimisen katsoa olevan huomaamatonta ja sen tapahtuvan usein kuin itsestään. Tämä ei kuitenkaan tarkoita, että oppija saavuttaisi parhaan potentiaalinsa pelkästään yksin opiskelemalla ja pelaamalla oppimispeliä. Oppimisen täyden potentiaalin kannalta tärkeää onkin myös ulkopuolinen *ohjeistus*, joka ohjaa oppijaa oikeaan suuntaan. Oppimista ei voida ulkoistaa oppimispelille, vaan itse oppimisen tukeminen vaatii aina jonkinlaisen ulkopuolisen ohjaajan.

5. Oppimispelit koulussa

Aiemmin tutkielmassa on käsitelty lähinnä oppimispelien rakentamista ja siihen liittyviä erilaisia teorioita. Väittäisin, että teoria on kuitenkin vain niin hyvä kuin sen lopullinen toteutus. Tämän vuoksi on tärkeää tutustua myös valmiisiin oppimispeleihin.

Oppimispelien käyttöä ja hyödyntämistä kouluissa ei pitäisi pelätä, vaan rohkeasti kokeilla niiden tehoa erilaisissa oppimistilanteissa, koska lapset ovat diginatiivien sukupolvea ja pelaavat mielellään. Diginatiivit ovat pienestä asti olleet ainakin jonkinlaisessa vuorovaikutuksessa digitaalisten ohjelmistojen, laitteiden ja pelien kanssa. Huomioitavaa on myös, että oppilaiden aikaisempien pelikokemusten, pelaamattomuuden tai asenteiden oppimispelejä kohtaan ei pitäisi olla esteenä pelien koulukäytössä, koska niillä ei havaintojen perusteella ole merkitystä lopulliseen oppimistulokseen [Hoblitz 2015].

Oppimispelejä onkin pyritty hyödyntämään erilaisilla tavoilla Suomen koulujärjestelmässä, eri-ikäisten lasten ja nuorten oppimisen tukemisessa. Tässä luvussa tarkastellaan kahta sisällöltään, tavoitteiltaan ja toteutukseltaan melko erilaista suomalaista oppimispeliä. Ensimmäiseksi tutustutaan jo jonkin aikaa tarjolla olleeseen Ekapeliin ja tämän jälkeen uudempaan Valopeliin.

5.1 Ekapeli – Peli opettaa lukemaan

Ekapeli on nykyään laajalti Suomen kouluissa käytössä oleva oppimispeli, jonka kohderyhmänä ovat alun perin olleet lukemaan opettelevat tai jonkinasteisesta lukihäiriöstä kärsivät lapset. Pelin kehitystyö on ollut pitkäaikainen Jyväskylän yliopiston ja Niilo Mäki Instituutin projekti. Nykyään pelistä on saatavilla monia eri versioita, jotka on kohdennettu esimerkiksi eskari-ikäisille ja ekaluokkalaisille tai jollain lukemisen osa-alueella harjoitusta kaipaaville lapsille. Pelistä on nykyään saatavilla versio myös lapsille, joilla on oppimisvaikeuksia matematiikassa,

versio venäjänkielisille maahanmuuttajataustaisille lapsille ja versio ruotsia äidinkielenään puhuville. Myös muun kielisiä versioita on kehitetty. [Lukimat 2016]

Ekapeli on ollut markkinoilla vuodesta 2004 lähtien. Tuolloin pelistä oli olemassa vain yksi versio, joka on korvautunut useammalla erikoistuneemmalla versiolla. Kehittäjän mukaan pelit mukautuvat pelaajan taitotason mukaan. Tämä tukee oppimisprosessia tarjoten yksilöllisen haasteen jokaiselle peliä pelaavalle lapselle. [Ekapeli esite 2011]

Vaikeustason mukautuvuus on oppimispelille tärkeää ja luvussa 4 käsiteltiin Kiilin [2005] ja Annettan [2010] yhtenevää käsitystä asiasta. Koska Ekapeli on kehitetty mukautuvuus silmälläpitäen, toteuttaa se molempien aiemmin esiteltyjen teorioiden perusajatuksen: jotta oppijan olisi mahdollisimman vaivatonta keskittyä pelin kulkuun ja tätä kautta oppia uusia asioita oppimispelin kautta, tulee tämän säilyttää ahdistuksen ja tylsistymisen välimaastossa oleva flow-tila. Kun peli osaa mukautua tähän vielä mahdollisimman huomaamattomasti, kyetään oppijan keskittyminen säilyttämään kiinteästi itse oppimispelin sisällössä.

Opettaja voi opetuksen tukena hyödyntää peliin sisäisesti rakennettuja taitojen arviointitehtäviä tai seurata pelaajan edistymistä erilaisista viuhkakuvioista ja tulostauluista. Peli on myös suunniteltu siten, että oppilas voi pelata peliä niin koulussa kuin kotona. Pelaajan tiedot ja suoritukset siirtyvät palvelimelle ja sitä kautta koulun koneelta kotikoneelle ja päinvastoin, joten oppilaan ei tarvitse pelata kahta erillistä peliä, vaan voi jatkaa siitä mihin edellisessä pelaamispaikassa jäi. [Ekapeli esite 2011]

Pelaajan innostusta pelaamiseen tuetaan Ekapelissä pelin interaktiivisuuden lisäksi myös erilaisilla pelillistämisen keinoilla. Esimerkiksi suoritettuaan riittävästi tehtäviä Ekapelin eskari-versiossa avautuu lapselle käyttöön erilliseen tarrakirjaan uusia ominaisuuksia tai erilliseen satukirjaan uusia satuja, joita peli lukee ääneen tavuttaen. [Ekapeli esite 2011]

Ekapeli on eskari-ikäisten lasten keskuudessa koettu varsin mieluisaksi ja havaintojen mukaan melko hyvin lapsen mielenkiintoa ylläpitäväksi. Pelaamisen aikana on luonnollisesti odotettavissa, että lapsi tekee virheitä eikä aina vastaa kaikkiin kysymyksiin ja tehtäviin oikein. Tämä ei kuitenkaan havaintojen perusteella vaikuta lapsen tarkkaavaisuuteen edes silloin, vaikka virheet kerääntyisivät pienelle ajanjaksolle. Pelaamismotivaatioon tällä oli vaikutusta osalla lapsista, mutta motivaatio palautui ennalleen melko pikaisesti tapahtuneen jälkeen. [Ronimus 2012]

Oppimistulosten suhteen Ekapeli on osoittanut olevansa toimiva. Lasten lähtötasoon suhteutettuna, oppimispeli parantaa kaikkien lasten osaamista ja pe-

laamisen jälkeen havaitaan kehitystä. Yhtenä osa-alueena Ekapelissä on kirjainten ja niiden ääntämisen harjoittelu. Esimerkiksi ennen pelaamisen aloittamista kohderyhmänä käytetyt lapset osasivat paljon vähemmän kirjaimia ja niiden ääntämistä kuin pelaamisen jälkeen. Oppimisen ja motivaation välillä ei Ekapeliä pelanneilla lapsilla havaittu olevan suoraa yhteyttä. [Ronimus 2012]

Ekapelissä käytetään erilaisia palkitsemisjärjestelmiä, eli pelillistämistä. Esimerkiksi yhdessä pelin versiossa on käytössä tehtäväkirja ja toisessa eläintarha. Tehtäväkirjan käyttö näytti heikentävän lapsen kiinnostusta tehtävien tekemiseen verrattaessa siihen, että tehtäväkirjaa ei esimerkiksi olisi ollut tarjolla. Ulkoinen palkitseminen siis söi lapsen sisäistä motivaatiota keskittyä lukemiseen liittyviin tehtäviin. Palkitsemisjärjestelmät eivät kuitenkaan heikentäneet lapsen motivaatiota itse pelin pelaamiseen. [Ronimus 2012]

Aiemmin mainitsin Ekapelin mahdollistavan pelin tallennusdatan synkronoinnin kodin ja koulun välillä, jolloin oppilas voi jatkaa pelin pelaamista samasta tilanteesta kuin mihin edellisessä paikassa jäi. Havaintojen mukaan lapset kokivat kuitenkin Ekapelin pelaamiseen paremmaksi ympäristöksi koulun, koska pelin koettiin olevan kuitenkin enemmän koulutehtävä kuin viihteellinen peli. Peliä arvioidaan myös olevan hauskempi pelata ryhmässä kuin yksin, jolloin viihdearvon toteutumiseen vaikuttaa se, onko pelaamisessa mukana myös opettaja ja muita oppilaita. [Ronimus 2012]

Ekapelin muokkautuvuus oppijan osaamisen mukaan on myös noussut esiin pelin käyttöön kohdistuneissa tutkimuksissa. Ronimus [2012] kertoo, että peli osaa muokata itseään helpommaksi, mikäli oppilaan osaaminen ei riitä tason suorittamiseen riittävällä määrällä yrityskertoja. Hän jatkaa, että tämä koettiin kuitenkin valvovien opettajien mielestä huonoksi asiaksi, koska heidän subjektiivisen näkemyksen mukaan oppilas ei oppinut opeteltavaa asiaa enää yhtä tehokkaasti kuin miten se olisi ollut mahdollista haastavammalla vaikeustasolla. Tämä oli hänen mukaansa erityisesti havaittavissa kouluympäristössä pelattaessa ja tutkimus ehdottaakin, että ainakin pienillä lapsilla (6–7-vuotiaat) optimaalisen haastavuustason löytämiseen vaikuttavat myös oppilaan motivaatio ja se, missä peliä pelataan. Tutkimus osoittaa, että vaikeustason mukautuvuus on onnistuttu rakentamaan peliin, mutta se toimii ehkä liiankin tehokkaasti antaen oppijalle liian vähän mahdollisuuksia onnistua vaativammalla vaikeustasolla. Pienille lapsille lienee edelliseen perustuen ongelmallisempaa löytää algoritmin avulla oppimistuloksen kannalta ihanteellisin vaikeustaso.

Järvisalon [2008] mukaan ensimmäistä, vuonna 2004 julkaistun Ekapelin versiota pelanneilla lapsilla oli pelistä hyviä kokemuksia niiden muutaman vuoden ajalta kuin he peliä pelasivat. Noin puolet pelaajista lopettivat pelaamisen

jossain vaiheessa, mutta lähes kaikilla tämä johtui riittävän taitotason saavuttamisesta lukutaidossa. Samoin koko aineistossa noin puolella esiintyi vielä lukivaikeuksia ja noin 39 prosentilla arvioitiin lukutaidon saavuttaneen ikäryhmälle tyypillisen tason. Järvisalo [2008] pohtii tutkimuksessa, että Ekapelin ensimmäinen versio olisi tarjonnut täydentävän oppimismuodon muiden perinteisten keinojen rinnalle. Myös Annetta [2010] toteaa, että opetusta ei ole ehkä edes järkevää pyrkiä ulkoistamaan oppimispelille, vaan peliä voi käyttää ennemmin muun opetuksen tukena. ”Muu opetus” voinee täten ulkoisena toimijana tukea oppimispelistä saatavia tuloksia, eikä päinvastoin.

5.2 Valopeli – Liikuntaa pelin ja leikin varjolla

On yleisesti tunnettu tosiasia, että lapset ja nuoret liikkuvat liian vähän. Tämä voi tuottaa erilaisia ongelmia kuten ylipainoa ja jopa aikuisikään heijastuvia fyysisiä tai psyykkisiä vajavaisuuksia.

Valopeli on Tampereen yliopistossa kehitetty uudenlainen oppimispelin muoto, joka opettaa, tai ehkä kuvaavammin innostaa liikkumaan. Peli on ensisijaisesti suunniteltu 9–10-vuotiaille koululaisille ja tarkoitettu käytettäväksi liikuntatunnilla opettajan valvonnassa. Peliä voidaan kuitenkin muokata sopivammaksi myös erityisryhmille tai eri-ikäisille lapsille. [Valopeli 2013]

Pelissä heijastetaan suurehkoon tilaan (esimerkiksi liikuntasali) erilaisia valoilmiöitä ja näiden avustuksella annetaan lapsille erilaisia liikunnallisia tehtäviä suoritettavaksi. Tehtävät koostuvat monipuolisista erilaisista liikuntasuorituksista, jotka eivät keskity pelkästään kuntopiirimäiseen peruskuntoon, vaan myös motoriikkaa ja koordinaatiota kehittäviin toimintoihin. [Valopeli 2013]

Hakulinen ja muut [2013] toteavat, että itse pelin tekninen osuus ei vaadi pelin loppukäyttäjiltä kovin suurta investointia, vaan pelin saa toimintavalmiiksi noin 1000€ summalla. Pelin tekniikka rakentuu kannettavasta tietokoneesta, kaiuttimista, langattomasta peliohjaimesta (Playstation Move), ja pienestä määrästä tietokoneohjattuja valaistusvälineitä liikuteltavassa kärryssä.

Pelin sisältö ei perustu pelkkään valon ja varjon leikkiin, vaan peli pohjaa erittäin vahvasti tarinankerrontaan, jota lapsen mielikuvitus ruokkii eteenpäin. Luonnollisesti myös valolla ja äänellä on suuri osuus ryhmän ja yksilön immersiivisen kokemuksen syntymisessä. [Hakulinen *et al.* 2013]

Pelihetki rakentuu yhdestä pääjuonesta ja siihen liittyvistä sivutehtävistä. Pääjuoni alkaa aina lyhyellä alustuksella, jossa esitellään pelin hahmot ja niiden tarkoitus [Keskinen 2015]. Peli ei itsenäisesti reagoi pelaajien toimintaan, vaan peliä hallitsee aina jokin ulkopuolinen ohjaaja, esimerkiksi opettaja. Ohjaajan tehtävänä on siirtyä pelin juonessa eteenpäin ja ohjata pelitilannetta sen mukaan kun pelaajat etenevät tehtävällisesti. Tutkimuksissa on havaittu, etteivät lapset

enää pelitilanteessa huomanneet ulkopuolista ohjaajaa, vaan tämän onnistui pysytellä taka-alalla häiritsemättömästi. Ulkopuolisen ohjaajan on myös mahdollista keskeyttää peli, mikäli hän havaitsee jonkin vaarallisen tilanteen syntymisen pelin aikana. [Hakulinen *et al.* 2013]

Peli on melko erilainen kuin muut oppimispelit, joita tyypillisesti pelataan näyttöpäätteellä. Tässä ratkaisussa peli toimii toki digitaalisesti, mutta maailma heijastetaan osallistujien ympärille. Peliä voidaan kuitenkin arvioida myös hieman mukaillen ns. perinteisillä peliheuristiikoilla, jotka on esitelty luvussa 3.

Esimerkiksi voidaan nostaa vaikkapa liitteestä 2 löytyvä heuristiikka "*GU2: Näytön ulkoasu on tehokas ja visuaalisesti miellyttävä*". Tätä heuristiikkaa ei voi luonnollisestikaan arvioida ainakaan oppijan näkökulmasta, koska oppija ei pelissä käytä näyttöä. Näkisin kuitenkin, että heuristiikan voi ajatella sillä tavalla, että näyttönä toimii koko pelitilana toimiva huone. Heijastettavat valot ja varjot voidaan näin arvioida näytöksi ja pohtia tätä kautta, että ovatko heijastetut tehosteet tehokkaat ja visuaalisesti miellyttävät.

Joitain heuristiikkoja voi kuitenkin arvioida suoraan sellaisenaan. Esimerkiksi liitteessä 1 esitellyt *1(d) tavoite*, *2(b) viihdearvo* ja *2(d) immersio* ovat esimerkkejä tästä. Nämä heuristiikat ovat enemmän subjektiivista kokemusta mittaavia kriteerejä, eivätkä ne sisällä erillistä laitteistovaatimusta.

Valopelin ensisijaisena tarkoituksena on tarjota vaihtoehtoinen ratkaisu niille lapsille, jotka eivät pidä liikuntatunnin tavanomaisista suorituksista. Lapsi saattaa pelätä esimerkiksi huomion keskipisteenä olemista vaativassa liikuntasuorituksessa tai olla muuten arka. Pelissä on tarkoituksena toimia ryhmänä ja se on rakennettu niin, ettei peli missään vaiheessa kannusta tai ruoki syrjintää [Keskinen 2015]. Peli ei myöskään missään vaiheessa aseta lasta muiden tarkkailtavaksi, vaan kaikki pelissä tapahtuva suoritetaan ryhmänä ja yhteistyönä. [Hakulinen *et al.* 2013]

Hakulinen *et al.* [2013] kertovat, että Valopelin lopullisena tarkoituksena ei ole aiheuttaa lapsessa behavioristisia muutoksia, vaan lähinnä tarjota mahdollisimman monelle lapselle hyviä kokemuksia liikunnasta ja ehkä näin rohkaista heidän liikkumistaan muissakin tilanteissa.

Valopeliä testaamassa olleet lapset pitivät pelistä erittäin paljon ja yli puolet arvioivat pelin parhaalla mahdollisella arvosanalla. Erittäin pieni osuus arvioi pelin neutraaliksi tai melko huonoksi. Reilu kolme neljästä oli sitä mieltä, että haluaisi liikkua samalla tavalla toistekin ja lähes sama osuus oli sitä mieltä, että valopeli oli mukavampi liikuntamuoto kuin normaali liikuntatunti. Ehkä suurin huomioitava asia on kuitenkin se, että ne oppilaat jotka eivät liikkuneet vapaa-ajalla kokivat liikunnan Valopelin avulla mukavammaksi kuin tavallisella liikuntatunnilla. [Hakulinen *et al.* 2013]

Valopeli on osoitus siitä, että oppimispelejä ja hyötypelejä voi käyttää myös muissa kuin perinteisiksi mielletyissä oppiaineissa. Liikuntatunti muuttuu Valopeliä pelattaessa täysin erilaiseksi kokemukseksi ja saattaa innostaa myös liikunnasta muuten piittaamattomia liikkumaan aktiivisemmin mukana.

6. Yhteenveto

Tutkielmassa tarkasteltiin oppimispelien toteuttamisprosessia ja sitä, miten onnistuneita oppimispelejä voidaan hyödyntää tukena lasten opetuksessa. Tärkeäksi asiaksi todettiin videopeliheuristiikat, joilla voidaan seurata pelin onnistumista niin kehityksen aikana kuin myös sen jälkeen. Vuorovaikutuksen puolesta huomio kiinnittyi erityisesti oppimispelin vaikeustasoon ja sen mukautumiseen oppijan taitotason perusteella. Tämä on tärkeää pelaajan flow-tilan ylläpitämisen kannalta.

Vaikka tutkielmassa kyettiin käsittelemään melko hyvin tärkeimmät oppimispelien rakentamiseen liittyvät asiat, jäi jotain kuitenkin käsittelemättä. Olisi ollut melko perusteltua myös esimerkiksi pohtia oppimisen teoriaa ja sitä, miten oppiminen oikeastaan tapahtuu. Tämä on aiheena kuitenkin ehkä vielä laajempi kuin oppimispelien kehitys, joten tutkielmassa päädyttiin keskittymään aiheeseen enemmän pelikehityksellisestä näkökulmasta. Huomiona voitaneen todeta, että hyvä oppimispeli vaatii luonnollisesti myös taustatietoa oppimisprosessista.

Aiheena oppimispelit on tällä hetkellä mielenkiintoinen, koska suuntaus kouluissa on ollut jonkin aikaa kohti digitaalisempaa koulua. Sama ilmiö on havaittavissa jollain tasolla jokaisella Suomen kouluasteella. Opettaja-lehdessä [2013] aiheesta kirjoitettiin positiiviseen sävyyn ja todettiin, että oppimispelit sopisivat koulutilojen pienten muutosten kautta erinomaisesti Suomen kouluihin, kunhan opettajia tuetaan sopivien pelien valinnassa eri oppiaineiden tarpeisiin.

Helsingin Sanomissa [2016] julkaistussa artikkelissa kerrotaan kuudesluokkalaisten kyynistymisestä koulua kohtaan. Samassa artikkelissa todetaan myös, että he kokevat koulun olevan liian vanhanaikainen, ja että koululla ei ole heille tarpeeksi annettavaa. Artikkelia varten haastateltu psykologian professori Katariina Salmela-Aro pitää tuloksia huolestuttavina, mutta mainitsee samassa yhteydessä, että ”Heidät (oppilaat) saataisiin ehkä innostumaan uudelleen, jos koulu käyttäisi monipuolisemmin digitaalista opetusta.”

Väittäisin, että oppimispelit tulevat olemaan yhä kiinteämpi osa tulevaisuuden koulua. Ympäröivän maailma uudistuessa, tulee myös koulun uudistua. Tätä ei pitäisi nähdä uhkana, vaan ennen kaikkea mahdollisuutena. Hyvin toteutettu oppimispeli voi tarjota paljon mielenkiintoisemman, mukaansatempaavamman, opettavaisemman ja immersiivisemmän kokemuksen kuin perinteinen

oppitunti luokkahuoneessa. Kukapa ei itse haluaisi virtuaalisesti vierailla antiikin Roomassa tai tutustua 1800-luvun New Yorkiin? Lähitulevaisuuden oppimispelit yhdistettynä lähitulevaisuuden teknologiaan voivat mahdollistaa tämän.

Lähteet

Wilfried Admiraal, Jantina Huizenga, Sanne Akkerman and Geert ten Dama. 2011. The concept of flow in collaborative game-based learning. *Computers in Human Behavior* 27, 2011, 1185–1194.

Leonard A. Annetta. 2010. The "I's" have it: A framework for serious educational game design. *Review of General Psychology*, 14(2), 105–112.

H. Chen. 2000. *Exploring web users' on-line optimal flow experiences.* Dissertation, Syracuse University.

Mihaly Csikszentmihalyi, 1990. *Flow: The Psychology of Optimal Experience.* Harper and Row.

Mihaly Csikszentmihalyi, 1997. *Finding Flow: The Psychology of Engagement with Everyday Life,* Basic Books.

H. Desurvire, M. Caplan and J. Toth. 2004. Using Heuristics to Evaluate the Playa bility of Games. In: *Proceedings of ACM Conference,* CHI 2004, Collection of Abstracts.

H. Desurvire and C. Wiberg. 2009. Game Usability Heuristics (PLAY) for Evaluating and Designing Better Games: The Next Iteration. In: *Proceedings of the 3d International Conference on Online Communities and Social Computing*: Held as Part of HCI International 2009. Springer-Verlag: 2009

Simon Egenfeldt-Nielsen, Jonas Heide Smith and Susana Pajares Tosca 2013. *Understanding Video Games: The Essential Introduction*, Routledge, 229–254.

Ekapeli lukemisen taitojen harjoitteluun – LukiMat. 2016. http://www.lukimat.fi/lukeminen/materiaalit/ekapeli Viitattu 26.2.2016.

Ekapeli esite. 2011. http://www.lukimat.fi/lukeminen/materiaalit/esitteet/esite-ekapelia-koskien/view Viitattu 26.2.2016.

Laura Ermi and Frans Mäyrä. 2005. Fundamental components of the gameplay experience: Analysing immersion. Online publications of University of Tampere Hypermedia Laboratory. Checked 25.1.2016, http://people.uta.fi/~tlilma/gameplay_experience.pdf

R.A. Finke, T.B. Ward and S.M. Smith. 1992. *Creative Cognition: Theory, Research & Applications*. The MIT Press.

Christina M. Finneran and Ping Zhang. 2003. A person–artefact–task (PAT) model of flow antecedents in computer-mediated environments. *Int. J. Human-Computer Studies 59,* 475–496.

Sebastian Deterding, Dan Dixon, Rilla Khaled and Lennart Nacke. 2011. From Game Design Elements to Gamefulness: Defining "Gamification". In: *Proceedings of the 15th International Academic MindTrek Conference - MindTrek '11.*

J. Ghani. 1991. Flow in human computer interactions: test ofa model. In: Carey, J. (Ed.), *Human Factors in Information Systems: Emerging Theoretical Bases.* Ablex Publishing Corp.

J.A. Ghani and S.P. Deshpande. 1994. Task characteristics and the experience of op timal flow in human–computer interaction. *The Journal of Psychology* 128, 381–391.

Jaakko Hakulinen, Markku Turunen, Tomi Heimonen, Tuuli Keskinen, Antti Sand, Janne Paavilainen, Jaana Parviainen, Sari Yrjänäinen, Frans Mäyrä, Jussi Okkonen and Roope Raisamo. 2013. Creating Immersive Audio and Lighting Based Physical Exercise Games for Schoolchildren. In: *Proceedings of the 10th International Conference on Advances in Computer Entertainment (ACE '13)*, 308–319.

Helsingin Sanomat. 29.3.2016. Liki puolet kuudesluokkalaisista on tympääntynyt kouluun – etenkin maahanmuuttajat uupuvat. http://www.hs.fi/kotimaa/a1459136043111. Viitattu 31.3.2016.

Anna Hoblitz. 2015. Gaming Experience as a Prerequisite for the Adoption of Digital Games in the Classroom? In: *Proceedings of DiGRA 2015: Diversity of play: Games – Cultures – Identities.*

Kai Huotari and Juho Hamari. 2012. Defining Gamification - A Service Marketing Perspective. In: *Proceedings of the 16th International Academic MindTrek Conference - MindTrek '12.*

Elisa Järvisalo. 2008. Mitä kuuluu Ekapelin ensimmäistä versiota ahkerasti pelanneille? *NMI-Bulletin,* 18, 4, 32–39.

Tuuli Keskinen. 2015. *Evaluating the User Experience of Interactive Systems in Challenging Circumstances.* Ph. D. Dissertation, School of Information Sciences, University of Tampere.

Kristian Kiili. 2005. *On Educational Game Design: Building Blocks Of Flow Design.* Ph. D. Dissertation, Tampere University of Technology - Pori.

Kristian Kiili, Sara de Freitas, Sylvester Arnab and Timo Lainema. 2012. The Design Principles for Flow Experience in Educational Games. In: *Proceedings of Virtual Worlds for Serious Applications, 78-91 - VS-GAMES'12.*

Elina M. I. Koivisto and Hannu Korhonen. 2006. Mobile Game Playability Heu ristics Nokia. In: *Proceedings of the 8th Conference on Human-Computer Interact tion with Mobile Devices and Services - MobileHCI '06.*

Elina M. I. Koivisto and Hannu Korhonen. 2007. Playability heuristics for mobile multi-player games. In: *Proceedings of the 2nd International Conference on Dig-ital Interactive Media in Entertainment and Arts - DIMEA '07.*

Martin Maguire. 2001. Methods to support human-centred design. HUSAT Re-search Institute. Loughborough University. Available online at http://www.idealibrary.com.

Alison McMahan. 2003. Immersion, Engagement, and Presence. A Method for Analyzing 3-D Video Games, In: M. J. P. Wolf and B. Perron (eds.) *The Video Game Theory Reader*, Routledge, 67–86.

Frans Mäyrä, Tanja Sihvonen, Janne Paavilainen, Hannamari Saarenpää, Anna-kaisa Kultima, Timo Nummenmaa, Jussi Kuittinen, Jaakko Stenros, Markus Montola and Jani Kinnunen, Antti Syvänen. Monialainen pelitutkimus. 2010 Teos: Sami Serola. Ote Informaatiosta. 306–349.

Opettaja-lehti. 17.5.2013. Enemmän pelejä, vähemmän piuhoja. http://con tent.opettaja.fi/epaper/20130517/20/index.html. 20–21 Viitattu 31.3.2016.

Janne Paavilainen ja Hannamari Saarenpää. 2009. Videopelit - käytettävyydestä pelattavuuteen. *Tietoasiantuntija*, 5.

Miia Ronimus. 2012. *Digitaalisen oppimispelin motivoivuus. Havaintoja Ekapeliä pe-lanneista lapsista.* Väitöskirja, Jyväskylän yliopisto.

L.K. Trevino and J. Webster. 1992. *Flow in computer-mediated communication.* Com-munication Research 19, 539–573.

Valopeli. 2013. http://www.uta.fi/sis/tauchi/mmig/projects/aktiivitilat/lii kunnalliset_pelit/liikunnalliset_pelit.html Viitattu 21.3.2016.

Liitteet

Liite 1: Play-Heuristiikat

Kategoria 1: Pelaaminen/pelattavuus

(a) Pelaamisen mukavuus/kestävyys
(b) Haaste/strategia/tahti
(c) Pelimaailman johdonmukaisuus
(d) Tavoitteet
(e) Pelaajien ja pelityylien erilaisuudet
(f) Pelaajien tunne hallinnasta

Kategoria 2: Viihdyttävyys, huumori, tunneperäinen immersio

(a) Tunneperäinen yhteys

(b) Viihdearvo
(c) Huumori
(d) Immersio

Kategoria 3: Käytettävyys ja pelimekaniikat
(a) Dokumentaatio/tutoriaali
(b) Tilanne ja pisteytys
(c) Peli tarjoaa palautetta
(d) Terminologia
(e) Pelaajan kuormitus
(f) Näytön ulkoasu
(g) Navigaatio
(h) Virheenehkäisy
(i) Tarina ja sen immersio.
[Desurvire and Wiberg 2009]

Liite 2: Mobiilipeliheuristiikat
Kategoria 1: Pelin käytettävyys
GU1: Audio-visuaalinen esitys tukee peliä
GU2: Näytön ulkoasu on tehokas ja visuaalisesti miellyttävä
GU3: Laitteen käyttöliittymää ja pelin käyttöliittymää käytetään sillä tavalla kuin niitä on tarkoitettu käytettäväksi
GU4: Osoittimet ovat nähtävissä
GU5: Pelaaja ymmärtää termistön
GU6: Navigaatio on yhtenäinen, looginen ja minimalistinen
GU7: Kontrollit ovat yhtenäiset ja yleisten tottumusten mukaiset
GU8: Kontrollit ovat kätevät ja mukautuvat
GU9: Peli antaa palautetta pelaajan toimista
GU10: Pelaaja ei voi tehdä peruuttamattomia virheitä
GU11: Pelaajan ei tarvitse muistaa asioita turhaan
GU12: Peli sisältää ohjeistuksen

Kategoria 2: Liikuteltavuus
MO1: Peli ja pelisessio voidaan aloittaa nopeasti
MO2: Peli mukautuu ympäristöön
MO3: Keskeytyksiin reagoidaan järkevällä tavalla

Kategoria 3: Pelattavuus
GP1: Peli tarjoaa selkeät tavoitteet tai tukee pelaajan luomia tavoitteita

GP2: Pelaaja näkee etenemisensä pelissä ja voi vertailla tuloksia
GP3: Pelaajat palkitaan ja palkinnot ovat merkitseviä
GP4: Pelaaja tuntee olevansa hallinnassa
GP5: Haaste, strategia ja tempo ovat tasapainossa
GP6: Ensimmäinen kokemus on rohkaiseva
GP7: Pelin tarina tukee pelattavuutta ja on merkitsevä
GP8: Pelissä ei ole itseään toistavia tai tylsiä tehtäviä
GP9: Pelaajat voivat ilmaista itseään
GP10: Peli tukee erilaisia pelaamistyylejä
GP11: Peli ei jumiudu paikalleen
GP12: Peli on yhtenäinen
GP13: Peli tarjoaa ortogonaalisen esineistön
GP14: Pelaaja ei menetä kovalla työllä ansaitsemaansa omaisuutta

Kategoria 4: Moninpeli
MP1: Peli tukee kommunikaatiota
MP2: On olemassa syyt kommunikaatiolle
MP3: Peli auttaa pelaajaa löytämään muita pelaajia ja pelejä
MP4: Peli tukee ryhmiä ja yhteisöjä
MP5: Suunnittelu minimoi poikkeavan käyttäytymisen syntymistä
MP6: Suunnittelu poistaa verkon vaikutuksen peliin.
[Koivisto and Korhonen 2006; 2007]

How usability components affect user navigation behaviour

Tobias Reinhardt

Abstract.
The evaluation of websites can be made, to some extent, by observing user interaction behaviour. Previous studies have identified many navigation metrics that predict user navigation success. However, there has only been little effort to describe the association between these metrics and website usability. In this thesis, I investigate how website usability might affect user navigation behaviour. Some potential correlation between navigation metrics and usability components are gathered. Potentially, these metrics can reveal usability issues. Furthermore, this thesis may strengthen some of the existing metrics that have been identified to correlate with user navigation success.

Keywords: user experience; browsing; information search; usability metrics.

1. Introduction

The internet contains more information, features and media than never before and is becoming the key medium to reach information. Web technologies are developing rapidly, and constantly more innovative navigation solutions are developed to serve the purpose of reaching the increasing amount of web objects. It is thus essential to be able to test whether users can conveniently reach the information they need using various tools available. If website navigation solutions fail to be usable, the user will be dissatisfied and is blocked from reaching knowledge [Fang & Holsapple 2010]. User frustration, disorientation and the lack of confidence have been identified to be common problems of hyperspace. These problems can result from a poor usability of a website [Levene 2010, 211–213].

User browsing behaviour has been widely analysed and has applications in wide variety of fields. For example, it is used in optimization of the web servers, web personalization and knowledge gathering. There has been however little focus on using user browsing behaviour to evaluate usability and only very recently some practical solutions have appeared in this field (e.g., the program LATTE by Thomas [2014]).

Improved usability of a website can drastically improve the success of user web interaction. The usability can be improved with various usability evaluations. The evaluations can be done with automatic usability evaluation tools or manually. Automatic usability analysis can cut the business costs of testing, and it suits well for websites since the data can be automatically retrieved through the network.

Some studies have attempted to discover user navigation metrics that would indicate user navigation success [Gwizdka & Spence 2006; Herder 2002; Thomas 2014; Smith 1996; McEaneaney 2001]. Other studies have then used these metrics to assess navigation structure or navigation tools [Van Oostendorp et al. 2009; Pardue et al. 2009; Fang & Holsapple 2010]. However, there are many complexities and constraints involving these metrics. Additionally, these studies do not consider how user navigation behaviour might develop over a longer period, and even some of these studies are contradictory. There are many variables in human web interaction. Thus, there is a great chance of using these metrics incorrectly. Furthermore, since there are no physical constraints in designing navigation structure for web both designing and navigating in websites are difficult tasks [Spence 1999].

The purpose of this thesis is to clarify the use of navigation behaviour assessing website usability. My hypothesis is that it is possible to some extent deduce website usability from user navigation behaviour. The usability definition given by Nielsen [1992, 26] is used. The definition divides the usability into measurable components which are learnability, efficiency, memorability, errors and satisfaction. I will confirm my hypothesis by inspecting the effect of each usability component alone on user navigation behaviour. The components are likely to accommodate various features that have been previously identified to affect user navigation behaviour. These features might be, for example, search complexity, user goal and motivation. I will relate each feature to a usability component and further review the influence on user navigation behaviour. This will hopefully reveal how each usability component affects user navigation behaviour. Additionally, I intend to explain why a certain usability component might correlate with certain user navigation metrics.

This said the thesis is divided into three main chapters investigating relationship betweens website usability and navigation metrics. A short insight into user web interaction and some potential navigation behaviour that indicate usability issues are given in Chapter 3. In Chapter 4, the effects of each usability component on user navigation behaviour is investigated. Finally, in Chapter 5, a conclusion is derived and an overall summary is given. Before delving deeper the following chapter provides motivation for this topic and introduces some of the potential benefits of using navigation behaviour to assess websites.

2. Motivation

Various metrics can be derived from user behaviour and some of these might indicate usability issues. The usability metrics are a way of evaluating usability. Especially analysing web usage statistics and behaviour patterns has emerged as a usability evaluation method [Matera et al. 2006]. Other ways of evaluating usability can be, for example, heuristical evaluation and cognitive walkthrough

[Matera et al. 2006]. Metrics have the benefits of being very comparable and to assess whether the usability of a product actually improved [Tullis 2008, 8–10]. Some of the web interaction can be implicit, unconscious interaction that goes unnoticed by users but is detected by interaction data, and the interaction data might reveal some usability issues [Atterer et al. 2006].

Usability metrics can be used in many usability evaluation methods (e.g., lab test, online test). Metrics can be used in laboratory environments to support user self-reported metrics and questionnaires. This method is vital in the development of early phases of the product. Later, the product can be fine tuned with online usability tests. This kind of testing applies especially to software and web products. Compared to laboratory tests online tests have the benefit of being able to test the vast amount of users which increase the confidence in the testing. These tests can be accompanied with short online questionnaires to give more insight into the products usability and what might be wrong. However, a test performed in the laboratory by individual level is still richer since the user can be directly observed. [Tullis 2008, 57–58.]

There has been some debate about the current metrics of user navigation behaviour being far from accurate and that subjective evaluation (e.g., questionnaires) is a far better usability indicator. Despite this, it does not make sense to use subjective evaluation in situations like automated usability evaluation or where adaptive navigation support is given to users whenever they are in trouble [Herder 2003; Thomas 2014]. Additionally, these metrics have gained more attention since the recent development in automated usability evaluation, machine learning and data mining techniques. The advancement in web log analysis and other methods of gathering user web interaction data help catch user navigation data. Machine learning can be used to associate rules between user struggling and their navigation behaviour. Data mining can be used to discover patterns and differences among user navigation. Thomas [2014] identifies two kinds of automatic web usability or users experience analysis. The one is where usability is deduced from static elements such as website structure and website loading speeds. The other is where the website is evaluated based on user behaviour, for example, to discover various associations between pages and user browsing patterns.

Some of the potential benefits of identification of problems from user navigation behaviour are listed below:

- It is nearly impossible for developers to consider every user interaction.
- Developers receive immediate feedback on the changes they make to the website.
- It is easy to observe thousands of users' web interaction sessions.
- Users work in their natural environment.

3. Describing user navigation

In this chapter user navigation and user web interaction are examined from a broader perspective. The purpose of this chapter is to gain a sense of user navigation and to reason that user navigation behaviour has the potential to reveal usability issues. I hope to achieve this by reviewing general user web usage. By explaining the difference between navigation and browsing. Finally, by reviewing two existing web interaction frameworks.

3.1. General search process

To understand the user navigation behaviour more precisely it is crucial to consider the general user behaviour in the hyperspace. Simultaneously, it is possible to reason the need for navigation and the necessity of such as navigation tools and navigation structure to the information search process. The hyperspace search process is unique and can't be directly compared to real world analogies. Hence, information search strategies that work in real life might not work efficiently in hyperspace, for instance, finding a document from a filing cabinet that is best found by considering the alphabetical order. Finding such document in hyperspace might involve a combination of query sentences and navigation.

The web provides two methods for information search: search engines and navigation tools. The user who performs information search session usually includes both of these methods to complement each other. A query search can be conducted either in the global environment (e.g., Google search engine) or in intranet environment (e.g., websites search bar). For example, the user might first execute a query and then pick relevant websites they will navigate. Then the user might change query syntax and navigate again. This iterative process can continue until the user is satisfied or gives up. The iterative characteristic is due to the web being an open environment. Thus, the user seldom knows precisely in advance what he or she might be searching. The navigation here plays a fundamental role in steering and assisting the user to more specific information while the query seldom results in an exact information the user needs. [Levene 2010, 9–42, 211–212.]

3.2. Navigation and browsing

The terms navigation and browsing are widely used together in the literature. Navigation and browsing are also sometimes used colloquially interchangeably. These terms are, however, defined differently. Additionally, the term browsing is frequently misused and misunderstood, and has many conflicting definitions [Pilgrim 2007]. We therefore explicitly define them and distinguish them from each other.

It is interesting to notice that while Smith [1996] used the term browsing to indicate user movement through web page links, McEneaney [2001] used nav-

igation instead of browsing to indicate the same user movement. Pilgrim [2007, 95–98] distinguished these two terms by noting that browsing is a mode of navigation when the website does not provide any supplementary navigation tools such as search tools, sitemaps or site indexes. When supplementary navigation tools are not available, the user has to fall back to conventional navigation mode called browsing where the user traverse through a website using menus and embedded links provided by the website. The user might first browse through the website before deciding to use any supplementary navigation tool. Pilgrim [2007] suggests that user navigation can embody different navigation strategies and that a navigation strategy is a sequence of navigation options such as the back button of the browser, some supplementary navigation tool (search index, sitemap, etc.) and websites embedded links or menus.

Spence [1999] defines web browsing through cognitive psychology and suggest that: "The activity of browsing is registration of content." He gives a real life example where a reader might take a rapid glance through morning newspaper before deciding what to read. By this example, the user also has the possibility to browse within a supplementary navigation tool. For example, the user can glance through a sitemap and choose an interesting site. Spence's [1999] framework in Figure 1 illustrates the relation between navigation and browsing.

Browsing is more of an opportunistic way of traverse through a website. The user tries various links and interprets the contents. Browsing affects the user's image of the system. However, the process is not random as the user is more likely to follow a link that they perceive to satisfy their needs most. Moreover, the strategy of browsing might change during sessions. The user might first start to browse website by an exploratory nature and to answer what useful the website can provide. The browsing can then become more directed where the user might browse towards a specific target. The term "weighted browsing" can be used for this and as the internal model develops the browsing is likely to become more weighted. [Spence 1999.]

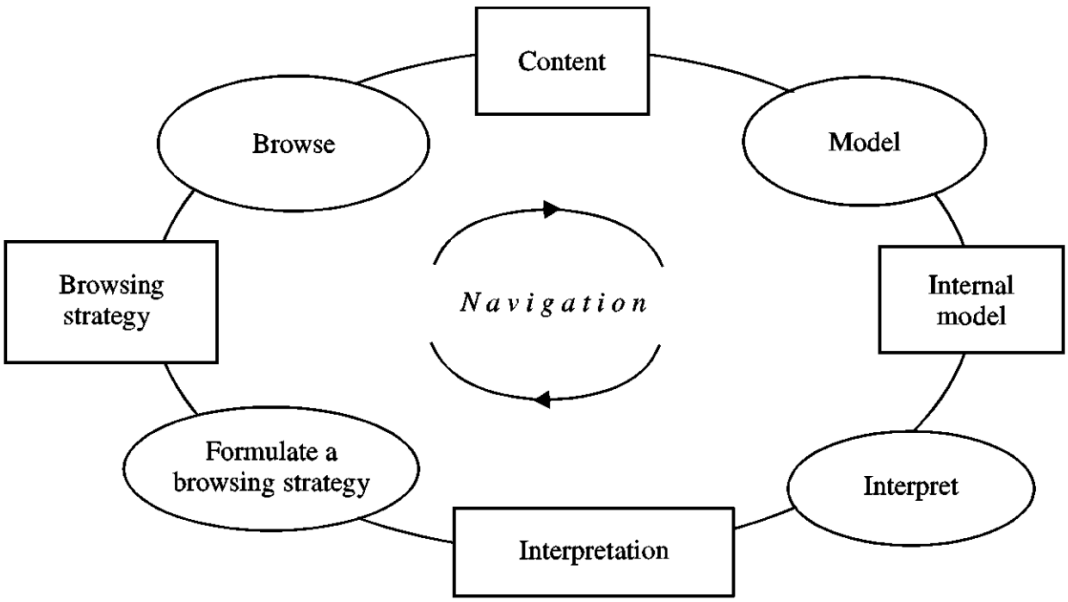


Figure 1: Spence's [1999] framework for navigation

Navigation is a wider term than browsing and browsing is a mode of navigation where content scanned in an opportunistic way. Navigation and browsing are defined as follows. Navigation is the process of orienting and path finding in an information space using various navigation tools. Browsing is the process of viewing a web content and selecting an interesting target from that content.

When referring to navigation behaviour every selection of web objects, choosing of navigation strategy and tool and forming and using of mental model are considered. Notice that the term navigation will incorporate browsing and will not consider the forming of query statement in search bars. Whereas when referring to browsing behaviour only selection of links is considered. Note that this will not consider, for example, switching between browser tabs or tools.

3.3. Web interaction frameworks

Many Web interaction frameworks exist to model various aspects of user interaction with websites. These frameworks are not a complete description of interaction. They can be used to reason about various design decisions and to infer new research directions and hypotheses [Pilgrim 2007, 74–75]. Two existing frameworks to give an insight into user navigation and browsing are introduced. They also help to reason about how and why components of usability might be related to some user features that affect navigation behaviour.

3.3.1. CoLiDeS

Kitajima et al. [2000] developed a model called CoLiDeS which is an acronym for Comprehension based linked model of deliberate search. It is based on text

comprehension models. The model explains the process of selecting a web object and concentrates on the aspect that screen objects compete for the users attention. Screen objects are divided into schematic (e.g., navigation bars and menus) and actual objects (e.g., pictures and hyperlinks). Actual objects are the interface elements user can click. The schematic objects are clustering of objects relating to each other. Schematic objects contains actual objects and can contain other schematic objects. The model consists of the following four phases:

- Parsing: Users do a bottom-up and top-down scanning on pages and associate visually related objects. Top down is "knowledge driven" scan where users expect certain schematic objects in the content. Bottom-up scan is "perception driven" where users build schematic objects by considering individual (schematic or actual) objects.
- Focusing on: While a user is parsing the user is easily drifted to focus on points or areas of interests which might be related to their goal. Focusing helps user comprehend smaller areas of a website.
- Comprehension: In this phase user elaborates, compares and interprets the meaning of objects. This stage is significantly facilitated by users domain knowledge and the knowledge of interface conventions.
- Selection: The user selects an object. The selection depends on how well the object can fulfil the user's goal. The selection depends on the degree of similarity between goal and object. The similarity depends on the literal matching of goal, representation of the object and the frequency user has used a path before to achieve a certain goal before.

The model helps to predict which objects a user is likely to interact with. The different objects draw the users attention and the attention is driven by user's goal. A successful search is considered to be the one where the user selects an object from the current disposable objects that seem to fulfil his/her goal most (selection) and this search strategy would satisfy user's goal in the end. This being said an error case is considered when the user has selected an object ends up in impasse where none of the objects seems to fulfil users goal. In such situation user usually refocus on a new area of content or he/she simply presses the back button of the browser. An error case is likely to arise, for example, from bad label naming or object being vague. [Kitajima et al. 2000.]

The model suggests that similarity of user traversed path to the optimal path and revisitation patterns would be potential metrics to indicate successful search. The indication is consistent with empirical studies by Herder [2003] and Gwizdka [2006]. The model does not explain anything about user satisfaction or frustration. Moreover, as noted by Smith [1996] user can spend time on websites just exploring and entertaining themselves with no exact particular goal.

From the model's point of view every web object contributes to user attention and the objects affect user navigation. Not only the navigation structure but the whole website affects user navigation.

3.3.2. Pilgrim's human web interaction framework

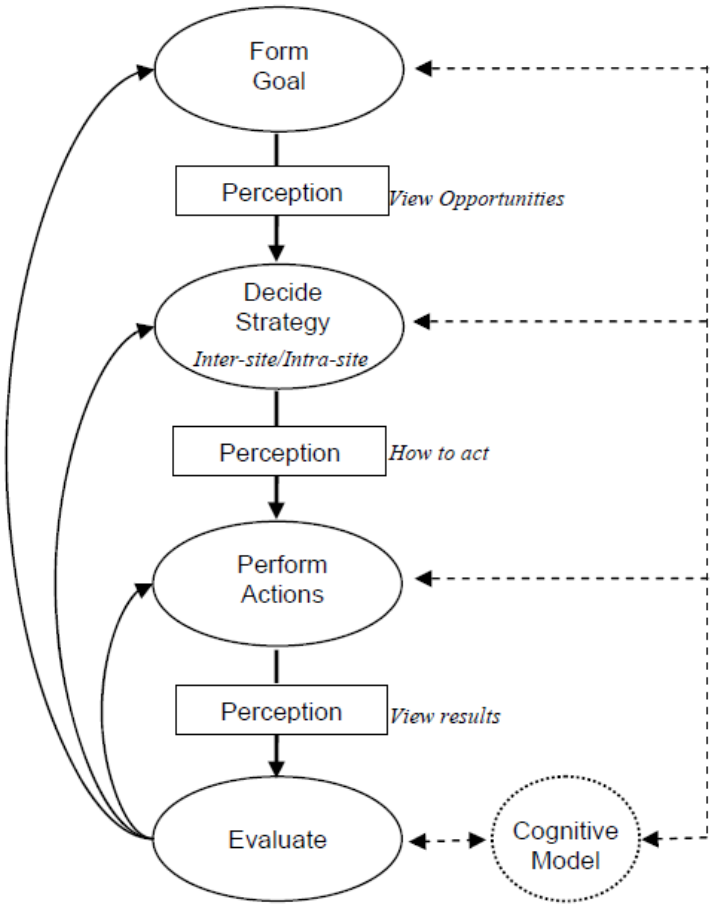


Figure 2: Web interaction framework proposed by Pilgrim [2007]

Pilgrim [2007, 71–87] reviews a set of human-computer interaction frameworks and identifies common stages in human web interaction. These are goals, strategy, action, perception, evaluation, cognitive model and feedback. He proposes a new framework which is illustrated in Figure 2. The framework composes of four stages which are forming of goal, deciding of strategy, performing actions and evaluation. The framework emphasises the role of perception and the process of choosing a navigation strategy. Through perception, the user can identify the available navigation strategies and the user choice of navigation strategy is greatly influenced by user preferred navigation strategy.

Perception is an ongoing process that is constantly present in each step of the framework. Through perception, each step is constantly being evaluated. User evaluates whether the chosen goal, strategy and actions will lead to the correct outcome. The user's cognitive model which is the image of the system is tuned or altered when user evaluates and reflects the outcomes of the interaction and compares it to the existing cognitive model. Each step, in turn, is influ-

enced by the cognitive model. The forming of a goal is highly affected by user's domain knowledge, the choice of strategy and actions are greatly influenced by previous experience, and the evaluation is influenced by the users' understanding of the world through the cognitive model. [Pilgrim 2007, 84–87.]

A goal can be defined as a state user wishes to achieve, for example, reaching a certain knowledge about something or by having to store data into a database. Pilgrim [2007, 88–94] reviews previous literature on user's goals which suggest that the user's goals have a level of specificity in the web environment. He proposes that a goal can land on a continuum of goal specificity which ranges from open to closed goal. He further divides the closed goal into a single goal or an aggregation of many goals. An example of an open goal could be by asking what the web page can provide and an example of a closed goal could be checking the deadlines of an internet course. Pilgrim [2007, 88–94] argues that users are likely to have open goals due to their lack of knowledge or when they arrive at a new and unknown web page. As the knowledge and image of the system increases the user is likely to be able to express the need more accurately and thus goal specificity increases from open to closed.

A strategy is a carefully devised plan to achieve a goal. In web navigation, it is a sequence of navigation options (back button key, menu list etc.) user chooses to use. Users might choose a set of tools to achieve a certain goal or if no tools are available, they can fall back to standard browsing where the user navigates through a web site using various links provided by the web page. It is useful to note that website can constrain the possible navigation strategies users can employ. There are many studies which try to identify patterns among strategies and it has been discovered that users' cognitive style has a significant impact on the chosen navigation strategy. [Pilgrim 2007, 96–108.]

4. Usability and navigation behaviour

Users' successful navigation on web page depends on their understanding where they are at the moment and where they can go from their current position. It is emphasised that users do not need to remember their location. They should be able to deduce their location from their perception. Therefore, navigation structure and correct representation of information both assist user navigation. Firstly, navigation structure enable users to travel between pages, present the relation between pages and communicate the content to users. Secondly, the correct presentation of information helps users understand where they are. If the presentation of information does not match the users mental model the tasks becomes difficult. [Garrett 2011, 118–127.]

The improvement of web page usability will facilitate easier information finding. Such increase in usability usually can be detected by an increase in the

user interaction success. The aim of this chapter is to illustrate how user navigation behaviour relates to usability.

The standard ISO 9241-11 defines the usability in the following way: "Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." The definition is a known standard. However, a more widely adopted and throughout definition is given by Nielsen [1992]. Nielsen [1992, 23–37] divides the usability into components which are learnability, efficiency, memorability, errors, satisfaction. This definition is used here.

Only few studies have attempted to correlate usability components directly with various navigation patterns. However, the literature has identified many features that affect user navigation behaviour. Thus, relating usability components to these features might reveal navigation behaviour that would correlate with certain usability component. By using this approach, it is possible to discover behaviour that would indicate usability. A list of features that might be affected by usability and have been identified to have a significant effect on user navigation behaviour are listed in Table 1. Each of the usability components is discussed separately in the upcoming sections.

Class	Feature
Individual differences	Prior knowledge
	Motivation
	Interest
	Emotion
	Cognitive load
Task characteristics	Task difficulty
	Task complexity
	Goal specificity
Contextual	Time constraints
	Interface

Table 1: List of features that have been identified to affect user navigation and which might also be affected by usability components. The features are gathered from the various sources. [Dorum 2012, 42–43; Juvina & van Oostendorp 2006; Fang & Holsapple 2010; Gwizdka & Spence 2006; Pilgrim 2007]

It is difficult to explain directly how usability affects user navigation behaviour. An easier approach might be to explain which characteristics are facilitated by usability that also have been identified to affect user navigation behaviour. These features present guidelines to design websites [Dorum 2012;

Swanson 2012; Pilgrim 2007]. Parunak [1989] found that navigational topologies support different navigation strategies [Marsh 1997, 17]. Pilgrim [2007] also noted that depending on the navigation tools the website might provide the website constraints the strategies users are able to perform. Figure 3 illustrates the directional association between usability and navigation metrics.

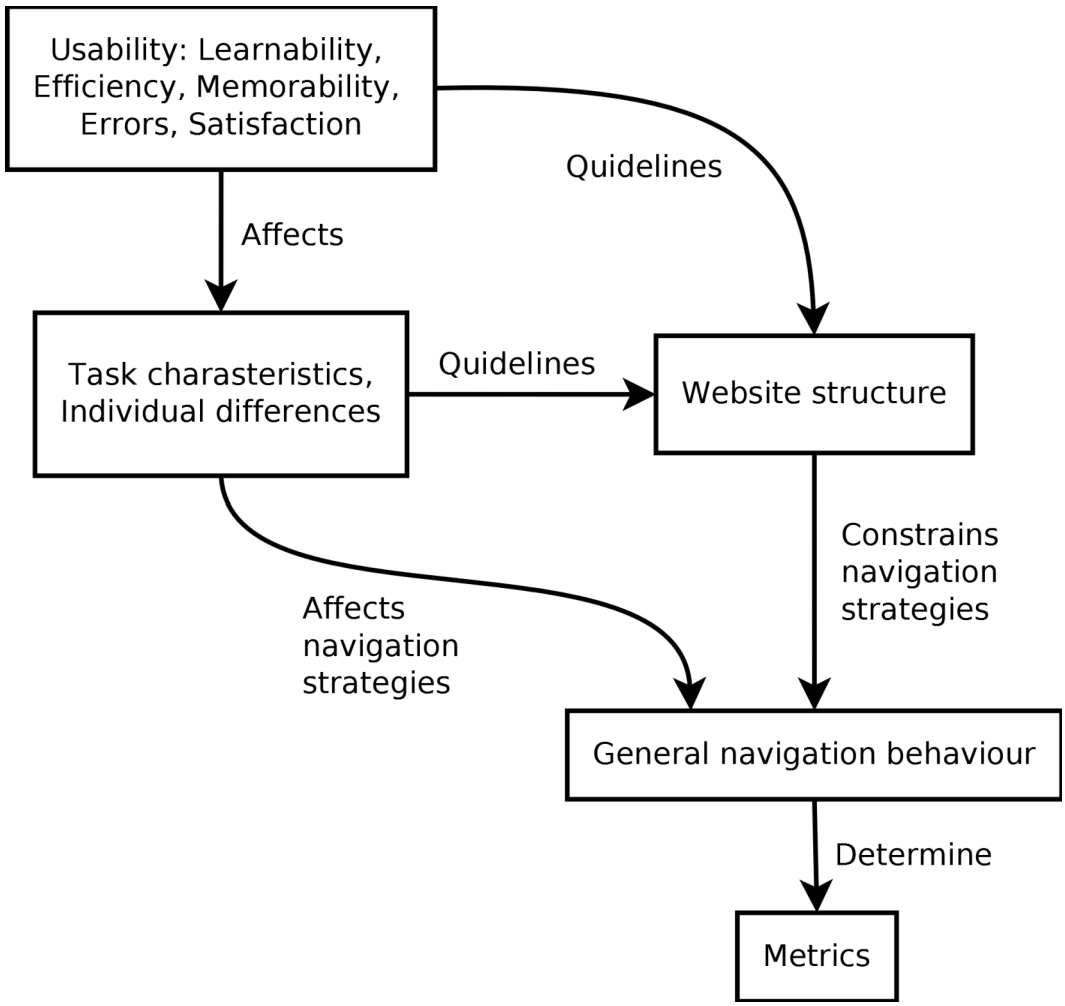


Figure 3: Proposed relation between navigation behavior and usability.

4.1. Learnability

Learning leads to the more efficient use of a system. The learnability is usually measured by having user performing a task in a certain time frame. The measuring should consider both time it takes for a user to master the system and time it takes to learn a certain degree to begin being productive. Note that users should not master the system completely to do some useful work. [Nielsen 1991, 27–30.]

In the case of navigation learnability would indicate that user quickly reaches a comprehension of the website structure and where information might reside.

Both user navigation and browsing cause incidental learning. [Swanson 2012, 24–27]. Learning might alter users' prior knowledge. Norman and Rumelhart [1976] classify learning into three categories which are the accretion of data, tuning of conceptual schemas, and restructuring of new conceptual schemas. A schema contains interrelationships of various components or objects. Which represent certain situations. A user uses these schemas to add new data to his/her knowledge. If the information does not fit into existing schemas the user may alter or create new schemas to comprehend new relationship or perspective of information. The definition of conceptual schemas is similar to conceptual frames introduced by Minsky [1974]. Frames are mental structures that are stored in memory which can be configured and fit reality. Frames can be, for example, used to predict various outcomes. Minsky [1974] gives a case upon entering a new room: A user might have an existing frame for that type of room where only a few variables such as door material or room size are accordingly configured after the user has inspected these elements.

Much of the real world navigation theories can be applied to websites. Thorndyke and Stasz [1980] elicit three stages users exploit in chronological order when they intend to learn new environment: First they learn spatial orientation, then they acquire route knowledge, and finally they begin to shift their frame of reference from ego centred to world centred frame of reference. [Swanson 2012, 32–33.]

All websites most likely differ from each other. A user is unable to predict where the information resides and what procedures are needed to reach his/her goal on a web site he/she has never visited before. For example, a user is unable to know whether a website even provides a menu to reach a specific location. Therefore, the user is unable to plan his/her strategy accurately. Thus, he/she can't accurately formalize his/her goal and consequently the goals have low level of specificity. However, as soon as the user arrives at the web site and begin to browse he/she start to build or alter their conceptual model of the site. This will result in a better formalisation of goals and generally increase the specificity of goals. The better the website communicates the structure to the users the better they are able to form conceptual model of the web site [Swanson 2012, 39–40]. Accurate conceptual model results in higher level of specificity of their goals.

McEaneaney [2001] and Smith [1996] both studied the success and disorientation of user navigation. These studies have however contradictory results. Smith [1996] concludes that a user with a linear path and lesser revisitation result in decreased disorientation. She assumes that disorientation results in a degradation of navigation performance. On the contrary, McEaneaney [2001] concluded that users who followed a shallow hierarchical navigation path, staying close to landmarks and revisits often pages had more success in their naviga-

tion. Gwizdka and Spence [2005] speculated that this contradiction may have resulted from users performing different task types.

The development in users' cognitive model increase the users ability to describe accurately their strategy of approach which in turn results in users employing a more direct navigation towards information. Thus, the user might browse deep towards the information and returning to landmark to employ a new carefully planned deep search towards needed information. Therefore, browsing path includes links deep from the navigation structure [Pilgrim 2007, 88–94, 105–108]. McEaneaney [2001] used "stratum" measure to indicate the degree of browsing path linearity. High path linearity has values closer to one and low path linearity has values closer to zero. Combining user goal specificity and "stratum" would result in users having planned search with have high path linearity and users having less planned search with low path linearity. This result is consistent with results from Gwizdka and Spence [2007] where "stratum" remained as a good predictor of user navigation success.

Besides the increase in goal specificity users also learn the various shortcuts and navigation tools (search bar, index lists, sitemaps, etc.) over time the website provides. When the users visit repetitively a website, they are likely to use these shortcuts and navigation tools to reach a destination with as few clicks as possible. In other words, they navigate deep into the structure with only very few clicks. If such a path was to be modelled it could be, for example, a ratio between navigation path length and the distance between path begin node and path end node in the actual structure)

4.2. Memorability

A memory schema can be modelled as an association of nodes and links inside brain. The concepts of memory and learning are both closely associated with each other. Memory and learning both assist each other. Recall and recognition help the user choose conceptual schema which assists in accretion of information, tuning and creation of new conceptual schemas. Whereas both the ease of learning and the better the website match user's conceptual schemas decrease user cognitive load which in turn assist the user to remember the website. [Norman & Rumelhart 1976; Minsky 1974; Fang & Holsapple 2010; Benjamin et al. 2014.]

Because of the tight interrelationship between memorability and learnability, the memorability also contributes to the user's ability to formalise goals. A more in-depth review about the formalization of goals has been given in Section 4.1 and Subsection 4.3.2.

Concept maps can be used to investigate how well a user has formed a conceptual model from a website [Swanson 2012, 37–40]. As a user has a clear map of the site he/she is likely to remember a page well (knowledge about where the

web objects reside within the page). Therefore, the user can be confident to return to the remembered page where he/she knows that he/she can start navigating to a certain direction. A landmark can be defined as a clear point from where a user can start navigating. Marsh [1997] argued that memorability of the website structure can be predicted from the number of visits to a node. McEneaney [2001] and Herder [2003] both agreed that user page revisitation correlates with user's success in navigation tasks.

Memorability is usually measured by user performance when he/she has not used the system in a while [Nielsen 1993, 31–32]. In the case of website interaction, memorability can span either current session or multiple sessions. In a current session, the user can memorise the pages he/she has visited. Thus, it is assumed that user's previously visited pages in a session are likely to affect her/his future navigation on the website. The previous usage of the website or other websites can increase and tune user's previous knowledge of an interface usage [Swanson 2012]. Therefore, users usually have expectations about interface elements and where they are may reside. Violating interface conventions can have impacts on user performance [Santa-Maria & Dyson 2008]. The knowledge about the interface is likely to decrease user's need to shift attention and search for a page object. As such, the knowledge is likely to result in more linear mouse movement and reduced scrolling especially scroll ups. Thomas [2014] found that struggling sessions had more user scroll ups than normal sessions.

4.3. Errors

Users' behaviour can go against their intention. The actions users make or omit against their intentions are regarded as user errors. Some errors are easier discoverable by users themselves than others [Nielsen 1993].

By nature simple slips are usually discovered and corrected rapidly by users themselves, for example, a user might accidentally click on a wrong link. In turn, mistakes can be harder to discover, for example, users might not accomplish their goals because they have devised a false plan of actions. Some of the most common causes of user errors are cognitive overload, distraction, time stress and faulty evidence. Systems should be designed to prevent users from making mistakes. [Norman 2013 162–217.]

According to CoLiDeS model (introduced in Subsection 3.3.1) browsing paths that diverge from paths which would satisfy user's goals can be considered errors. One reason to such a "wrong path," can be poor link labelling. Another reason can be that a user is unable to follow an optimal information scent because of too much distraction or information overload. After choosing a "wrong path," the user is likely to reach an impasse where none of the web objects satisfy user's information scent. After reaching an impasse, the user has

several possible actions. Firstly, he/she might backtrack using back button of the browser or by links. Secondly, he/she might shift attention to another part on a site. Finally, he/she might use a tool (page index, menu list, etc.) to select a different site from the websites hierarchy. [Kitajima et al. 2000.]

Besides a user choosing a wrong path due to bad link labelling, vague links or wrong information scent the user might also choose a wrong path due to an incorrect strategy of approach or by accidentally to click links he/she had no intention clicking on. The user might accidentally click on a wrong button due to the buttons of a menu residing too close to each other and due to inaccurate motor skills. Such errors are especially an issue on mobile phones. The user usually rapidly recovers from such errors by pressing back button of the browser. Whereas a wrong strategy can easily go unnoticed by users but are likely to lead to an impasse.

The navigation behaviour caused by errors are likely to result in an increasing number of circles and the overall pages visited. "Circling" is defined as a retracing of steps and reviewing of earlier pages [Thomas 2014]. The overall pages visited increase due to the "circling" and visits to irrelevant pages. Users countering an impasse might spend an excessive amount of time shifting focus and searching for objects that might satisfy their information scent. The number of pages visited and time on pages can be however deluded by users motivation of exploring the website (This is examined more depthly in Section 4.5). Thomas [2014] discovered that struggling sessions tend to have more circles and time spend on each page. Herder [2003] also found median view time to correlate with disorientation. Users with difficulties tended to spend more time on each page.

Users' cognitive style can significantly delude the effect of errors on navigation behaviour. Some users might employ sporadic navigation where the user visits many pages and spents less time on each page. Sporadic navigation is due to their cognitive style. There is nothing wrong in possessing a certain cognitive style. Though, there have been findings showing that users with certain cognitive style are less successful than others in web search. [Kinley et al. 2014.] Rest of this section will examine how cognitive style affects user navigation.

Cognitive style describes an individual preferred way of processing information. Riding [1991] develops a known cognitive style analysis which places individuals in an orthogonal wholist-analytic (W-A) and verbal-imagery (V-I) dimensions. "The wholist-analytc dimensions of cognitive styles describes the habitual way in which people think about, view, and structure information in wholes or parts." Wholists retain the situations as whole while analysts retain the situations as a collection of parts. "Verbal-imagery dimension of cognitive styles describes an individual's tendency to process information either in words (verbal) or mental pictures." Imagers tend to represents the information by

mental pictures. Whereas verbalizers tend to associate information through word or verbal representations. [Kinley et al. 2014.]

Kinley et al. [2014] investigated the relationship between individuals placed in wholist-analytic, verbal-imagery dimensions and their web interaction behaviour. They derived a relationship model from their empirical study which is illustrated in Figure 4.

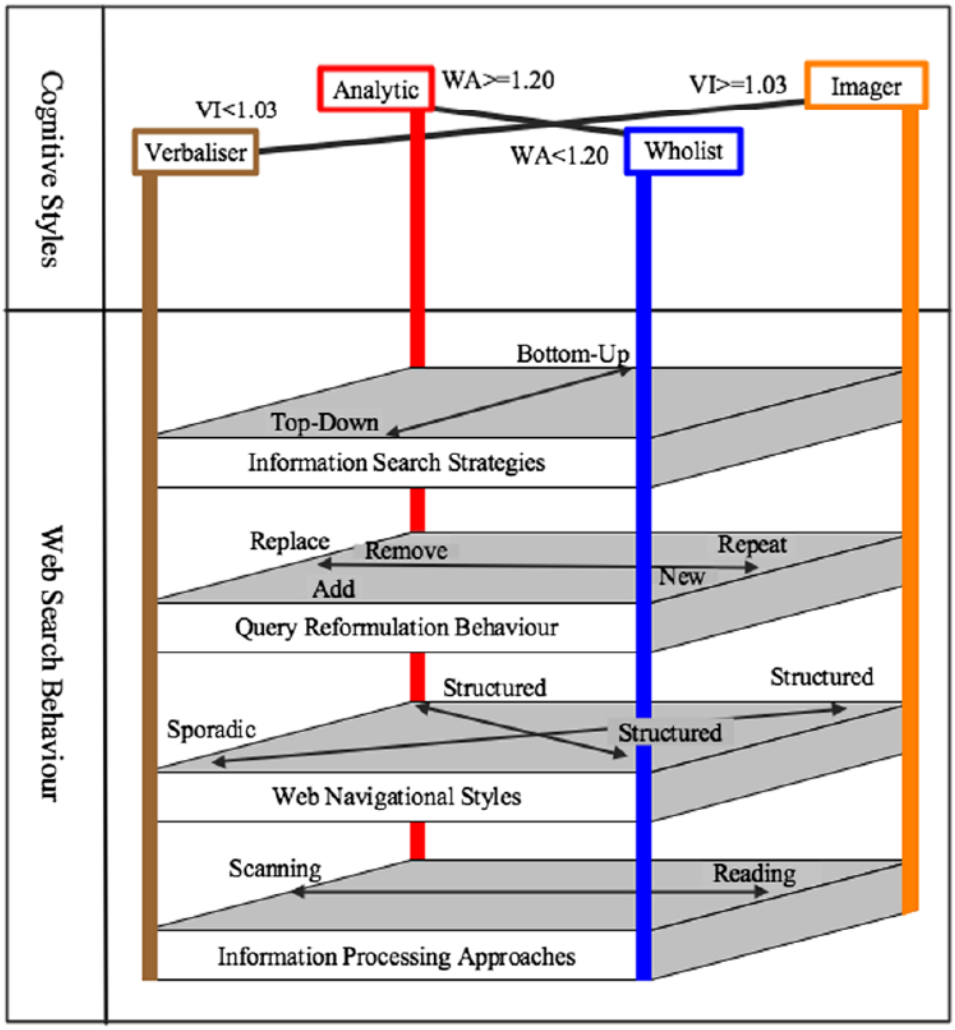


Figure 4: Model of correlations between navigation style and cognitive style devised empirically by Kinley et al. [2014].

Verbalisers tends to use more sporadic navigation style than analytics, imagers and wholists who tend to use more structured navigation style. Moreover, verbalisers and analytics prefer scanning over reading while imagers and wholists prefer reading. Scanning refers to a style of browsing where the page is looked in more general manner. In this style of navigation, individuals are likely to switch rapidly between topics and pages. Whereas in reading individuals go through pages comprehensively. In reading style, individuals are likely to spend more time on page and visit lesser pages. Individuals who follow sporadic navigation style scans multiple pages and often returns to change

their query formulation. They tend to visit more pages and used browsers back button more often than the individuals following structured navigation style. The individuals who follow this structured navigation style moved through pages and formulated their queries more carefully. They tend to spend more time on each page and use more navigation buttons available than their counterparts. [Kinley et al. 2014.]

4.4. Efficiency

While users use the system they learn it. At some point, a user reaches the top of the learning curve and becomes expert system user. This is when the efficiency of the system can be properly measured. The time it takes to learn the system is not considered when measuring efficiency. The efficiency is usually evaluated by measuring the time it takes to perform the task. [Nielsen 1993.]

The website encourages exploration and people usually just browse to find entertainment. They might not have a certain specific task in mind. Thus, it is not always sensible to measure efficiency in certain tasks. However, there are task types where the performance can be measured such as, form filling and fact finding.

Various navigation tools (sitemaps, menu lists, search bars, etc.) help the user navigate to a certain location as quickly as possible. Then again if the website has too many tools and shortcuts the site might increase user's cognitive load and the user can become disoriented. Many researchers use the assumption of disorientation introduced by Wood [1986] who assumes that the disorientation in users will decrease their search performance. Therefore, a dynamic balance between the amount of tools should be considered. While the tools can increase efficiency, they can also decrease it.

The purpose of tools and shortcuts are to help users understand the structure and create short paths to often visited locations. Gwizdka and Spence [2006] found that similarity to the optimal path is a good indicator of user's navigation success and task difficulty. In other words, if the user can follow the shortest paths to their destinations they are likely to succeed. It is usually the case that shorter paths in navigation structure are faster for the user to travel than those that are longer. Thus, the similarity of users' navigation paths to the actual optimal path is likely to correlate with efficiency.

4.5. Satisfaction

A user might no be able to find the required information or gather knowledge that would satisfy him/her. The user might not even understand the structure of the website. In these cases, the user is likely to switch to another website [Levene 2010]. Thomas [2014] uses the term "swapping" to describe the behaviour where a user navigates from website to global search engine to find a new website. He then suggests that web interaction sessions which included swap-

ping had more struggling compared to sessions which had lesser swapping. In other words, if the user swaps after a short period of time he/she is likely to struggle on the website. User struggling is likely to result in user dissatisfaction and frustration with the website. As user constantly encounters impasses and can't find the information they struggle they are likely to switch page. This is consistent with the concept of focus switching introduced by CoLiDeS model. Here the switching of focus is just considered to happen at a more global level of web interaction.

The number of swaps is closely related to time spent on a website. As soon as the user begins to struggle, he/she is likely to discontinue to navigate on the site.

5. Conclusion

This thesis described general user web interaction behaviour and introduced two existing web interaction frameworks. The description allowed for an insight into user navigation behaviour and initiated a starting point to investigate user navigation behaviour. My initial hypothesis was to show that website usability can be evaluated through user navigation behaviour. Our approach was to explain the relationship between usability and user navigation behaviour. My method was to examine each component of usability and explain the affected metrics through theories and features influenced by these components. This method revealed metrics that were affected by some usability components. Only the clear, strong influences were considered. A summary of the gathered correlations between components and metrics are given in Figure 5. The components are highly interrelated and the components might overlap together with some metrics. It is also worth noticing that some of the metrics are not independent of each other. The literature suggests that patterns of metrics could be better indicators than single metrics.

This thesis elicited a set of hypotheses which requires more in-depth analysis and empirical evaluations. There are certainly many more metrics that could have gone unnoticed by this thesis. Neither the scope nor the space of this thesis could cover all the possible metrics. Some of the associations between usability and navigation metrics were clarified. This thesis suggests that the evaluation of website using user navigation data seems very possible. However, the review does not reveal the efficiency compared to other usability evaluation methods.

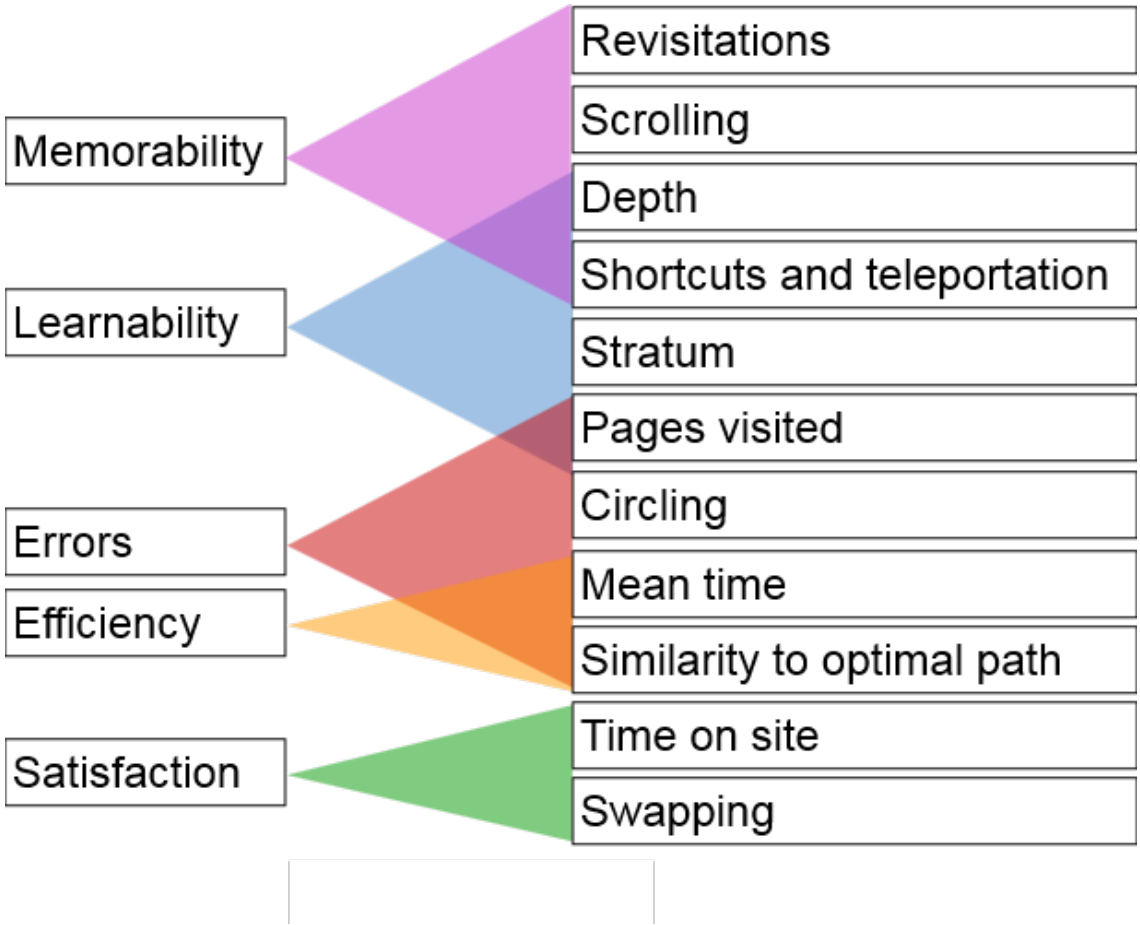


Figure 5: Proposed possible correlations between usability components and navigation metrics.

Bibliography

Atterer Richard, Wnuk Monika and Schmidt Albrecht. 2006. Knowing the user's every move – User activity tracking for website usability evaluation and implicit interaction. In: *Proceedings of the 15th International Conference on World Wide Web*. 203–212.

Campbell Donald. 1988. Task Complexity: A Review and Analysis. *The Academy of Management Review* 13, 40–52.

Dorum Kine. 2012. *The Role of Individual Differences and Metaphor in Hypermedia Navigation*. Ph D. Dissertation, College of Medicine, Biological Sciences and Psychology School of Psychology, University of Leicester.

Fang Xiang and Holsapple Clyde. 2010. Impacts of navigation structure, task complexity, and users domain knowledge on Web site usability—an empirical study. *Information Systems Frontiers* 13, 453–469.

Garrett Jesse. 2011. *The Elements of User Experience (2nd Edition)*. Peachpit Press.

Gwizdka Jacek and Spence Ian. 2005. Indirect assessment of web navigation success. In: *Extended Abstracts on Human Factors in Computing Systems*, 1427–1430

Gwizdka Jacek and Spence Ian. 2006. What can searching behaviour tell us about the difficulty of information tasks? A study of web navigation. In: *Proceedings of the 69th Annual Meeting of the American Society for Information Science and Technology.*

Herder Eelco. 2003. Revisitation patterns and disorientation. In: *Proceedings of the German Workshop on Adaptivity and User Modelling in Interactive Systems*, 291–294.

Juvina Ion and van Oostendorp Herre. 2006. Individual differences and behavioural metrics involved in modeling web navigation. *Universal Access in the Information Society* 4, 258–269.

Kinley Khamsum, Tjondronegoro Dian, Partridge Helen and Edwards Sylvia. 2014. Modelling users' web search behaviour and their cognitive styles. *Journal of the Association for Information Science and Technology* 65, 1107–1123.

Kitajima Muneo, Blackmon Marilyn H and Polson Peter G.2000. A Comprehension-based Model of Web Navigation and Its Application to Web Usability Analysis. In: McDonald Sharon, Waern Yvonne, Cockton Gilbert. In: *Proceedings of HCI 2000*. 357–373

Levene Mark. 2010. *An Introduction to Search Engines and Web Navigation (2nd Edition)*. Wiley.

Liu Peng and Li Zhizhong. 2013. Task complexity: A review and conceptualization framework. *International Journal of Industrial Ergonomics 42, 553–568.*

Marsh Anna Virginia. 1997. *Structure and Memorability of Web Sites*. Master's thesis. Department of Mechanical and Industrial Engineering. University of Toronto.

Matera Maristella, Rizzo Francesca and Carughi Giovanni Toffetti. 2006. Web usability: Principles and evaluation methods. *Web Engineering*, 143–180.

McEneaney John. 2001. Graphic and numerical methods to assess navigation in hypertext. *International Journal of Human Computer Studies* 55, 761–786.

Minsky Marvin. 1974. A framework for representing knowledge. *Technical Report. Massachusetts Institute of Technology*, Cambridge.

Nielsen Jakob. 1993. *Usability Engineering*. Morgan Kaufmann Publishers Inc.

Norman Donald. 2013. *The Design of Everyday Things Revised and Expanded Edition.* Basic Books.

Pardue John Harold, Landry Jeffery Paul, Kyper Erik and Lievano Rodrigo. 2009. Look-ahead and look-behind shortcuts in large item category hierarchies: The impact on search performance. *Interacting with Computers* 28, 235–242.

Pilgrim Chris. 2007. *User Goals and Web Site Navigation- Implications for the Design of Sitemaps*. Ph D. Dissertation, Faculty of Information and Communication Technologies, Swinburne University of Technology.

Riding Richard. 1991. Cognitive styles analysis. *Birmingham, UK: Learning and Training Technology*.

Rosman Benjamin, Ramamoorthy Subramanian, Mahmud Hassan and Kohli Pushmeet. 2014. On user behaviour adaptation under interface change. *Proceedings of the 19th International Conference on Intelligent User Interfaces.* 273–278.

Rumelhart David and Norman Donald. 1978. Accretion, tuning and restructuring: Three modes of learning. In: *Semantic Factors in Cognition*, 37–53.

Santa-Maria Luis and Dyson Mary. 2008. The effect of violating visual conventions of a website on user performance and disorientation. How bad can it be? In: *Proceedings of the 26th Annual International Conference on Design of Communication.*

Smith Pauline. 1996. Towards a practical measure of hypertext usability. *Interacting with Computers* 8, 365-381.

Spence Robert. 1999. A framework for navigation. *International Journal of Human-Computer Studies* 51, 919–945.

Swanson Kari Gunvaldson. 2012. *Learnable Interfaces – Leveraging Navigation by Design*. Ph D. Dissertation, University of Washington.

Thorndyke, P. W. & Stasz, C. (1980). Individual differences in procedures for knowledge acquisition from maps. Available: http://www.dtic.mil/cgibin/GetTRDoc?AD=ADA065040&Location=U2&doc=GetTRDoc.pdf.

Tidwell Jenifer. 2010. *Designing Interfaces (2nd Edition).* O`Reilly Media.

Tullis Tom and Albert Bill. 2008. *Measuring the User Experience Collecting, Analyzing, and Presenting Usability Metrics.* Elsevier Inc.

Van Oostendorp Herre, Madridb R. Ignacio and Puerta Melguizo Mari Carmen. 2009. The effect of menu type and task complexity on information retrieval performance. *The Ergonomics Open Journal* 2, 64-71.

Wood Robert. 1986. Task complexity: definition of the construct. *Organizational Behavior and Human Decision Processes*, 37, 60–82.

Musiikin sisältöpohjainen haku ja tunnistaminen

Emmi Siitonen

Tiivistelmä

Musiikin sisältöpohjainen hakeminen on musiikin etsimistä sisältönsä perusteella joko nauhoitettua tai hyräiltyä näytettä käyttämällä. Musiikkia ei voida lähteä suoraan vertaamaan toiseen kappaleeseen, vaan se täytyy ensin saada muutettua helpommin luettavaan muotoon, esimerkiksi symboliseksi dataksi. Osa musiikin piirteistä on mahdollista muuttaa helposti merkkijonomuotoon, ja tämän jälkeen hakemisessa on mahdollista hyödyntää monia pitkälle kehittyneitä, normaalisti tekstin haussa käytettäviä, algoritmeja.

Musiikin sisältöpohjaiseen tunnistamiseen liittyy monia ongelmakohtia, joista osa on musiikin haulle yksilöllisiä. Niitä varten on kehitetty monia algoritmeja, joista jotkin ovat täysin uusia ja jotkin vanhoista kehitettyjä. Esimerkiksi näytteen taustamelun ja kohinan sekä hyräillyn näytteen poikkeamat ja tyylivirheet ovat keskeisiä ratkaisua vaativia ongelmia musiikin sisältöpohjaisessa hakemisessa.

Tämä tutkielma on kirjallisuuskatsaus musiikin sisältöpohjaiseen hakuun sekä sen peruspiirteisiin ja yleisimpiin ongelmakohtiin. Tutkielman tarkoitus on tutustuttaa lukija aiheeseen ja kiinnittää tämän huomio kaikista oleellisimpiin osa-alueisiin.

Avainsanat ja -sanonnat: sisältöpohjainen haku, likimääräinen sovittaminen, musiikillisten piirteiden erottaminen, QBH, audion sormenjälki.

1. Johdanto

Nykyajan kehittynyt teknologia on tuonut käyttäjien saataville valtavan määrän erityyppistä multimediadataa. Kuvia ja tekstiä voidaan hakea tarkasti ja kattavasti vain paria hakusanaa käyttämällä, ja näissä hauissa käytetyt hakualgoritmit ovatkin huippuunsa hiottuja. Äänen ja varsinkin musiikin hakuun vastaavia algoritmeja on kehitelty jo parisen vuosikymmentä, ja tarve näille algoritmeille on jatkuvasti kasvava. Siinä missä kuvia voidaan hakea antamalla hakukoneelle esimerkkikuva, voi käyttäjä etsiä myös kappaletta antamalla hakukoneelle näytteen etsitystä kappaleesta. Käyttäjä saattaa esimerkiksi kuulla radiosta mielenkiintoisen kappaleen tai hänen päässään saattaa soida kappale, jonka nimen tai esittäjän hän haluaisi saada selville. Tällöin haussa käytetty syöte voisi olla joko nauhoitettu ääninäyte kappaleesta itsestään tai käyttäjän itse hyräilemä tai laulama katkelma.

Yleensä haun tarkoitus on nimenomaan tunnistaa haettu kappale, mutta toisinaan käyttäjä saattaa myös haluta tietää kaikki kappaleet, jotka ovat hyvin

samanlaisia kuin haussa käytetty kappale, esimerkiksi plagiarismin havaitsemisen helpottamiseksi. Tämä tutkielma keskittyy kuitenkin pääsiassa musiikin tunnistamiseen, vaikka jotkin esitellyt algoritmit voivat hyvin soveltua myös samankaltaisuuksien havaitsemiseen. [Thomas et al. 2012]

Kokenut musiikin kuulija voi erottaa kappaleesta hyvinkin tarkasti eri säveliä ja sointuja kuulemansa perusteella, mutta koneellisesti sama ei onnistu yhtä helposti. Musiikin analysointi ja tunnistaminen perustuu useimmiten siihen, että audiodata muutetaan aluksi helpommin analysoitavaksi symboliseksi dataksi. Musiikin muuttaminen *suoraan* symboliseksi dataksi on kuitenkin ongelma, jota ei olla saatu ratkaistua, mutta sitä varten on kehitelty monia erilaisia tekniikoita [Laaksonen 2015]. Kun audiodata on saatu muutettua symboliseksi dataksi, se tallennetaan tietokantaan, josta sitä voidaan analysoida helposti vertailemalla muihin kappaleisiin esimerkiksi merkkijonohauissa käytettyjä algoritmeja soveltamalla.

Musiikin tunnistamiseen liittyy myös muita ongelmakohtia, jotka täytyy ottaa huomioon musiikin tunnistamiseen käytettäviä algoritmeja valittaessa ja kehitettäessä. Algoritmien täytyy esimerkiksi nopean ja tarkan hakutuloksen lisäksi olla kykeneviä mahdollisimman tarkkaan hakuun annetun syötteen laadusta huolimatta. Toisin sanoen syötteen taustalla kuuluva kohina, lauletun syötteen virheet tai huono äänenlaatu eivät saisi vaikuttaa haun tulokseen, vaan oikea hakutulos täytyisi saada tuotettua, vaikka etsitty kappale ja käytetty näyte eroaisivatkin toisistaan jonkin verran.

Tämä tutkielma on kirjallisuuskatsaus musiikin tunnistamiseen ja siihen liittyvien ongelmien ratkaisemiseen kehitetyistä tekniikoista ja algoritmeista. Toisessa luvussa kerrotaan musiikin tunnistamisesta ja siihen liittyvistä ongelmakohdista yleisellä tasolla, kun taas kolmannessa luvussa keskitytään enemmän musiikkiin ja sen tärkeimpiin piirteisiin. Neljäs luku käsittelee lyhyesti tietokantojen indeksointia ja viidennessä luvussa käsitellään kappaleiden vertailua eli varsinaisen hakutuloksen aikaansaamista. Koska aihe on laaja ja kehitettyjä tekniikoita ja algoritmeja suuri määrä, on tämä tutkielma vain pintaraapaisu aiheeseen. Tutkielman tarkoitus on tutustuttaa lukija aihealueeseen ja kiinnittää tämän huomio kaikista tärkeimpiin asioihin, kuten algoritmien joustavuuteen ja virheiden sietokykyyn.

2. Musiikin tunnistamisesta yleisesti

Musiikkia on mahdollista hakea erilaisia hakusanoja, kuten esittäjän nimeä, käyttämällä muun muassa hakupalvelu Googlen avulla, mutta on olemassa monenlaisia tilanteita, joissa käyttäjällä ei ole antaa hakukoneelle minkäänlaista hakusanaa. Tällöin on tärkeää, että kappale voidaan hakea pelkän sisältönsä perusteella esimerkiksi nauhoitettua näytettä käyttämällä. Tällaista hakuja

kutsutaan *sisältöpohjaisiksi hauiksi* (content-based search). Normaalisti kyse on tilanteesta, jossa käyttäjä kuulee esimerkiksi radiosta kappaleen, jonka nimeä ei tiedä ja haluaa selvittää sen mahdollisimman nopeasti ennen kuin kappale ehtii loppua. Muun muassa *Shazam* on vuonna 2002 julkiseen käyttöön avattu, musiikin tunnistamista varten kehitetty sovellus, joka on erittäin tehokas sovellus tunnistamaan kappaleita kovankin melun seasta [Wang 2006].

Hausta, jossa käytetään käyttäjän itse tuottamaa näytettä, oli se joko laulaen, hyräillen tai viheltäen tuotettu, käytetään nimitystä QBH (query-by-humming) eli *hyräilyllä haku*. Hyräillen tuotetun näytteen käyttämistä on tutkittu jo vuonna 1995, kun Ghias ja muut ehdottivat tapaa mallintaa musiikin melodiaa ja sävelkorkeuksien eroja merkitsemällä niitä symboleilla "*U*" (up), "*D*" (down) ja "*S*" (same). Symbolit yksinkertaisuudessaan kuvastivat, oliko luettu sävel korkeampi, matalampi vai sama kuin edellinen. Vaikka tutkimusta pidetäänkin hyvänä lähtökohtana hyräiltyjen näytteiden hyödyntämiseen, on sen todettu myös olevan aivan liian karkea lähestymistapa monimutkaisen melodian kuvaamiseen [Hsu et al. 1998].

QBH-hakua muistuttava nimi on keksitty myös hauille, joissa ääninäytteenä käytetään esimerkiksi radiosta nauhoitettua pätkää, eli kun kyseessä on niin sanottu *esimerkillä haku* eli QBE (query-by-example). QBE-haut voidaan jakaa kahteen alaluokkaan: *tarkalla esimerkillä haku* eli QBEE (query-by-exact-example) ja *coveriversiolla haku* eli QBCE (query-by-cover-example). Ensimmäisessä näyte on täysin sama kuin etsitty kappale, kun taas toinen voisi olla vaikka eri soittimilla soitettu tai muuten hiukan muunneltu versio etsitystä kappaleesta.

Koska on olemassa tilanteita, joissa haussa käytetty syöte saattaa olla hyräilty, vihellelty tai laulettu katkelma etsitystä kappaleesta, oikean hakutuloksen saaminen tällaisissa tilanteissa on yleensä vaikeampaa kuin jos annettu syöte olisi suora nauhoitus kappaleesta itsestään. Tämä johtuu puhtaasti siitä, että vain harvat ihmiset kykenevät hyräilemään kappaleita ilman minkäänlaisia virheitä. Pelkästään tempo saattaa hyräiltäessä helposti joko kaksinkertaistua tai puolittua [Hu and Dannenberg 2002]. Samalla tavalla kappaleiden välisiä eroavaisuuksia lisäävät myös näytteen taustamelu, huono äänenlaatu ja kohina. Ääninäytteistä siis löytyy toisinaan kohtia, jotka ovat täysin ylimääräisiä tai turhia hakemisen kannalta. Tällaiset kohdat voivat yksinkertaisimmillaan olla vain hiljaisuuksia, eivätkä nekään saisi vääristää oikean hakutuloksen aikaansaamista.

Yllämainitun kaltaisia ongelmia voidaan ratkoa hyödyntämällä niin sanottua *likimääräistä sovittamista* (approximate matching). Toisin sanoen algoritmeja kehitetään niin, että ne pystyvät tunnistamaan kappaleet samanlaisiksi, vaikka niissä olisikin jonkin verran eroavaisuuksia.

Likimääräisen sovittamiseen on kehitetty monenlaisia eri toteutustapoja, ja nykyään se on yksi musiikin tunnistamisen perusominaisuuksista.

Likimääräisen sovittamisen mahdollistamisen lisäksi hakualgoritmien täytyy olla mahdollisimman tehokkaita, jotta etsitty kappale löytyisi laajasta tietokannasta mahdollisimman nopeasti ja vaivatta. Tietokanta täytyy olla rakennettu niin, että kaikkien kappaleiden läpikäynnin sijaan haku voitaisiin suorittaa siten, että vain mahdollisimman pieni osa tietokannasta tarvitsee käydä läpi. Yksittäinen kappale sisältää itsessään jo suuren määrän dataa, ja joskus yhdessä tietokannassa saattaa olla jopa miljoonia kappaleita. Esimerkiksi vuonna 2006 Shazamin tietokanta sisälsi yli kolme miljoonaa kappaletta, ja sen jälkeen määrä on oletettavasti vain kasvanut [Wang 2006]. Kappaleiden kustannustehokas säilöminen ja nopea läpikäyminen sellaisenaan onkin mahdotonta. Sen sijaan hakua on mahdollista nopeuttaa huomattavasti generoimalla jokaiselle tietokannan kappaleelle mahdollisimman pieni, jopa vain joidenkin kilotavujen kokoinen yksilöivä sormenjälki. Varsinkin mobiililaitteiden yhteydessä on erityisen tärkeää kiinnittää huomiota liikkuvan datan määrään ja esimerkiksi haussa käytetyn näytteen ei tulisi koskaan tarvita olla yli kymmentä sekuntia pitkä.

3. Musiikilliset piirteet ja niiden erottaminen

Ihminen kykenee erottamaan musiikista monenlaisia tunteita, kuten pirteyttä ja surumielisyyttä, ja kokenut musiikin kuuntelija jopa erillisiä sointuja ja säveliä. Koneellisesti saman aikaansaaminen ei vielä tänäkään päivänä täysin onnistu. Jotta musiikkia pystyttäisiin analysoimaan koneellisesti, täytyy siitä ensin erottaa erilaisia piirteitä, joita voidaan sitten muuttaa sellaiseen muotoon, jota koneen on helpompi lukea.

Musiikilla on hyvin omanlaisensa rakenne, joka koostuu monista eri piirteistä. Hsun ja muiden [1998] mukaan musiikilliset piirteet voidaan jakaa kolmeen alaluokkaan:

Staattinen informaatio. Kappaleen oleellisimmat piirteet, kuten sävellaji, tahti ja tempo.

Akustiset piirteet. *Kuuluvuus* (loudness) , *sävelkorkeus* (pitch), kesto ja kirkkaus. Nämä piirteet saadaan luettua suoraan musiikkiobjektien datasta.

Temaattiset piirteet. Teemat, melodiat, rytmit, harmonia, sävelet sekä soinnut. Temaattiset piirteet voidaan esittää merkkijono muodossa. Esimerkiksi ”C-Am-Dm-G7” on neljästä soinnusta muodostettu merkkijono.

Näistä kolmesta alalajista kuulijalle merkittävimpiä piirteitä ovat temaattiset piirteet. Esimerkiksi akustiset piirteet ovat yleensä niin sanotulta normaalilta kuulijalta piilossa ja liian vaikeasti havaittavia, kun taas muun muassa melodiat ja rytmit ovat selkeästi erottuvia piirteitä niin kokeneille kuin

kokemattomille musiikin kuulijoille. Akustisia piirteitä ja erityisesti sävelkorkeutta hyödyntämällä on toki aikaansaatu useita toimivia musiikintunnistustekniikoita, mutta ne toimivat parhaiten lähinnä nauhoitettuja näytteitä käytettäessä. Temaattisten piirteiden käyttäminen on varmempi lähestymistapa, kun haussa käytetään käyttäjän itse tuottamaa näytettä, sillä käyttäjän on helpompi tuottaa temaattisia piirteitä.

Teema on yksi tärkeimmistä musiikin yksilöivistä piirteistä, ja populaarimusiikin kertosäkeet ja klassisen musiikin motiivit ovat hyvin tyypillisiä teemoja. Melodia ja harmonia muodostavat ison osan teemasta, ja ne ovat näin ollen itsekin hyvin oleellisia osia musiikkia. *Melodia* on useista eri *sävelistä* koostuva sävelkulku, ja se esitetään yleensä niin sanottujen *säveltapahtumien* (note event) joukkona, jossa jokainen tapahtuma *(t,p)* koostuu kahdesta osasta: aloituspisteestä *t* ja sävelkorkeuden arvosta *p*. Aloituspisteellä tarkoitetaan tässä tapauksessa kohtaa, jossa sävel alkaa, sekunteissa mitattuna. *Harmonia* taas koostuu yhtä aikaa soivista erikorkuisista äänistä eli *soinnuista*, ja melodiaa muistuttaen se voidaan esittää sointumuutosten joukkona. Tässä joukossa jokainen tapahtuma *(t,s)* koostuu aloituspisteestä *t* ja soinnun symbolista *s*. [Laaksonen 2015]

3.1 Soinnut ja niiden rakenne

Soinnut siis muodostuvat joko kolmesta tai useammasta samanaikaisesti soivasta sävelestä, ja Choun ja muiden [1996] mukaan ne soveltuvat hyvin musiikin analysointiin, sillä ne sietävät hyvin virheitä. Toisin sanoen, jos säveliä ”do-mi-sol” kuvattaisiin soinnulla C, niin pienistä muutoksista huolimatta säveliä ”do-mi-mi-sol-sol” kuvattaisiin myöskin soinnulla C. Tämä johtuu siitä, että molemmat tapaukset yhä kuulostavat samalta, soinnulta C. Sointujen virheiden sietokyky koskee sävelten tuplaantumista, lisääntymistä ja pois jäämistä.

Chou ja muut esittelevät myös tavan, jolla yhtä aikaa soivista sävelistä voidaan koneellisesti muodostaa sointuja. Käytännössä kyse on siis sointujen tunnistamisesta. Aluksi täytyy vain määrittää kuvan 1 tapaan jonkinlainen sointujoukko, joka kattaa eri soinnut mahdollisimman hyvin. Tämän jälkeen tarvitaan vain muutama välivaihe, joiden perusteella luetut sävelet yhdistetään johonkin sointuun:

1. Etsi soinnut, jotka sisältävät eniten käyttäjän syötteestä luettuja säveliä.
2. Säästä kaikista lyhyimmät soinnut
3. Säästä soinnut, joiden juurisävel esiintyy ehdokkaiden joukossa useimmin.
4. Säästä soinnut, joiden viidesosasävel esiintyy ehdokkaiden joukossa useimmin.

5. Säästä soinnut, joiden kolmasosasävel esiintyy ehdokkaiden joukossa useimmin.

Tässä kohtaa riittää tietää, että juurisävelellä tarkoitetaan soinnun ensimmäistä säveltä, ja viidesosa- ja kolmasosasävelillä tarkoitetaan soinnun toista ja kolmatta säveltä. Esimerkiksi soinnun C juurisävel on ”do”, viidesosasävel ”mi” ja kolmasosasävel ”sol”.

C1: do	C2: do, mi	C: do, mi, sol	C7: do, mi, sol, si
D1: re	D2: re, fa	D: re, fa, la	Dm7: re, fa, la, do
E1: mi	E2: mi, sol	Em: mi, sol, si	Em7: mi, sol, si, re
F1: fa	F2: fa, la	F: fa, la, do	F7: fa, la, do, mi
G1: sol	G2: sol, si	G: sol, si, re	G7: sol, si, re, fa
H1: la	H2: la, do	Hm: la, do, mi	Hm7: la, do, mi, sol

Kuva 1. Sointujen tunnistamista varten muodostettu sointujoukko

Useimmiten välivaiheista läpi tarvitsee käydä vain pari ensimmäistä kunnes jäljellä on enää vain yksi sointu, joka valitaan haun tulokseksi. Esimerkiksi jos käyttäjän syötteestä saadaan luetuksi ääntä, joka sisältää sävelet |do, mi, mi|, niin ensimmäisen säännön mukaan valituksi tulevat soinnut ”C2”, ”C”, ”C7”, ”F7”, ”Am” ja ”Am7”. Tämä siksi, koska kaikki nämä kuusi sointua sisältävät molemmat sävelet ”sol”, ”mi”. Tämän jälkeen säännön kaksi mukaan näistä kuudesta soinnusta jäljelle jätetään ne, joiden pituus on kaikista pienin. Nyt jäljelle jää enää sointu C2, joka on myöskin haun tulos.

3.2 Hierarkkisuus ja toistuvat mallit

Jonesin [1974; Hsua ja muita [1998] lainaten] mukaan musiikkikappaleet koostuvat rakenteesta, johon liittyy kaksi perussääntöä: *hierarkkisuussääntö* (hierarchical rule) ja *toistuvuussääntö* (repetition rule). Hierarkkisuussäännön mukaan kappaleet koostuvat *liikkeistä*, liikkeet *virkkeistä*, virkkeet *lauseista* ja lauseet *kuvioista*. Näistä kappaleelle saadaan muodostettua hierarkkinen rakenne. Toistuvuussäännön mukaan taas jotkin nuottijaksot eli *aiheet* toistuvat useasti liikkeen sisällä. Tällaisia moneen kertaan musiikin sisällä toistuvia jaksoja kutsutaan myös *toistuviksi malleiksi* (repeating pattern), ja niitä pidetään yhtenä musiikin peruspiirteistä. Lähes jokainen kappale koostuu malleista, jotka toistuvat kappaleen sisällä useita kertoja, joskus samanlaisina, joskus hiukan muuteltuina. Sama sääntö pätee, oli kyse sitten klassisesta musiikista tai populaarimusiikista, ja kertosäteet, motiivit ja teemat ovatkin kaikki oikeastaan eräänlaisia toistuvia malleja. Etsimällä kappaleen kaikki toistuvat mallit ja valitsemalla niistä pisimmät ja useimmiten esiintyvät kappaleesta saadaan löydettyä sille omaperäisiä ja tärkeitä piirteitä, esimerkiksi pätkä kertosäettä.

Vertailemalla näitä omaperäisiä piirteitä tietokannasta löytyvien kappaleiden piirteisiin (toistuviin malleihin), voidaan etsitty kappale löytää tietokannasta ilman, että vertailuja tarvitsee tehdä koko kappaleen matkalta.

Kaikista intuitiivisin tapa lähteä etsimään kappaleen S toistuvia malleja olisi ensin generoida kaikki S:n alimerkkijonot ja verrata sitten jokaista alimerkkijonoa X S:än, jotta X:n esiintymiskertojen lukumäärä saataisiin laskettua. Kaikessa yksinkertaisuudessaan tämä lähestymistapa on kuitenkin myöskin erittäin tehoton, koska S:n pituuden ollessa *n* sen alimerkkijonojen lukumäärä on $n + (n-1) + \ldots + 1$. Alimerkkijonolle X, jonka pituus on *m*, täytyisi tehdä $O(mn)$ vertailua, jotta X:n esiintymiskertojen määrä S:ssä saataisiin laskettua. Pahimmassa tapauksessa kokonaiskompleksisuus olisi jopa $O(n^4)$. [Hsu et al. 1998]

Tällaisen ”brute-force”-haun tilalle onkin kehitelty erilaisia tapoja nopeuttaa toistuvien mallien hakua. Hsun ja muiden mukaan yksi tapa tehostaa toistuvien mallien etsimistä on käyttää *korrelaatiomatriisia*. Matriisin elementtien arvot kuvastavat löytyneiden mallien pituuksia, ja matriisin läpikäyminen on helppoa ja vaivatonta. Seuraava esimerkki havainnollistaa korrelaatiomatriisin käyttöä, ja sen tarjoamia etuja.

Olkoon S tutkittava kappale ja sitä vastaava melodiamerkkijono ”Bb5-Ab5-Ab5-Db5-C6-Ab5-Db5-Bb5-Ab5-Ab5-Db5”. Löytyneitä, epätriviaaleiksi malleiksi kutsuttuja merkkijonoja on tässä esimerkissä yhteensä kolme kappaletta, ”Bb5-Ab5-Ab5-Db5”, ”Db5” sekä ”Ab5”. Tämä siksi, että epätriviaaleiksi toistuviksi malleiksi *ei lasketa* malleja, jotka esiintyvät *ainoastaan* toisten mallien alimerkkijonoina. Toisin sanoen, koska merkkijono ”Ab5-Ab5-Db5” esiintyy ainoastaan merkkijonon ”Bb5-Ab5-Ab5-Db5” alimerkkijonona, sitä ei lueta omaksi mallikseen. Ainoastaan toisten merkkijonojen alimerkkijonoina esiintyvät merkkijonot tunnistaa siitä, että niiden frekvenssi on sama kuin niiden merkkijonojen, joiden sisällä ne esiintyvät. Tämän takia ”Ab5”, joka esiintyy merkkijonossa yhteensä 5 kertaa, eli useammin kuin ”Bb5-Ab5-Ab5-Db5” ja täten muuallakin kuin alimerkkijonona, luetaan omaksi toistuvaksi mallikseen.

Koska merkkijonossa S on 11 nuottia, luodaan toistuvien mallien hakuun 11x11 kokoinen matriisi T (kuva 2). Matriisin alkiosta, joka on rivillä i ja sarakkeella j käytämme merkintää $T_{i,j}$. Matriisia täytetään rivi kerrallaan, ja jokaisella rivin sarakkeella sen hetkisen rivin säveltä S_i verrataan sen hetkisen sarakkeen säveleen S_j. Niiden ollessa samat ($S_i = S_j$) solun arvoksi asetetaan 1, kuten ensimmäisen rivin soluun $T_{1,9}$. Sen lisäksi, jos $S_i = S_j$ ja $T_{i-1,j-1} \neq 0$, $T_{i,j}$:n arvoksi asetetaan $T_{i-1,j-1} + 1$. Tällaisessa tilanteessa merkkijonosta S on löytynyt toistuva malli. Esimerkiksi kuvassa 2 solut $T_{1,8}$, $T_{2,9}$, $T_{3,10}$ ja $T_{4,11}$ muodostavat neljän mittaisen merkkijonon, joka esiintyy melodiassa useammin kuin kerran.

On syytä huomata, että korrelaatiomatriisi kertoo vain löytyneiden mallien pituuden, mutta ei niiden frekvenssiä.

	Bb5	Ab5	Ab5	Db5	C6	Ab5	Db5	Bb5	Ab5	Ab5	Db5
Bb5	-							1			
Ab5		-	1			1			2	1	
Ab5			-			1			1	3	
Db5				-			1				4
C6					-						
Ab5						-			1	1	
Db5							-				1
Bb5								-			
Ab5									-	1	
Ab5										-	
Db5											-

Kuva 2. Korrelaatiomatriisi joka osoittaa toistuvien mallien pituudet

Matriisin täyttämisen jälkeen seuraava askel onkin löytää matriisista kaikki toistuvat kuviot ja niiden frekvenssit. Käytämme tähän niin kutsuttua *ehdokasjoukkoa* eli CS:ää (candidate set), jossa CS:n jokainen elementti on muotoa (kuvio, frekvenssi, alim_frekvenssi), missä frekvenssi ja alim_frekvenssi tarkoittavat kuvion esiintymistiheyttä ja sitä, kuinka monta kertaa kuvio esiintyy jonkin toisen kuvion alimerkkijonona. Alim_frekvenssin avulla saadaan tietää, esiintyykö kuvio ainoastaan muiden kuvioiden alimerkkijonona.

CS lasketaan matriisin jokaiselle ei-tyhjälle solulle seuraamalla neljää ehtoa, joiden yksityiskohdat sivuutetaan tässä tutkielmassa (ks. [Hsu et al. 1998]). Tämän jälkeen lasketuista ehdokasjoukoista poistetaan ne joukot, jotka esiintyvät ainoastaan jonkin muun merkkijonon alimerkkijonoina. Tämä onnistuu valitsemalla kaikki joukot, joiden Alim_frekvenssi > 0 ja joiden frekvenssi on sama kuin sillä joukolla, jonka alimerkkijonona kyseisen joukon merkkijono esiintyy. Jäljelle jääneet ehdokasjoukot ovat nyt (”Ab5”, 10, 2), (”Db5”, 3, 1) ja (”Bd5-Ab5-Ab5-Db5”, 1, 0). Lopuksi jokaiselle jäljelle jääneelle toistuvalle kuviolle lasketaan todelliset frekvenssit kaavalla

$$f(kuvio) = \frac{1+\sqrt{1+8\times frekvenssi}}{2}.$$

Nyt kuvioiden ”Ab5”, ”Db5” ja ”Bd5-Ab5-Ab5-Db5” todellisiksi frekvensseiksi saadaan kaavan avulla 5, 3 ja 2.

Hsun ja muiden [1998] testeissä korrelaatiomatriisi tekniikan avulla yksittäisen kappaleen kaikki toistuvat kuviot löytyvät suurella onnistumisprosentilla ja hyvinkin nopeasti. Esimerkiksi 600 nuotin kappaleesta kaikki toistuvat mallit löytyvät vain kahdessa sekunnissa, joskin 1000 nuotin kohdalla aikaa kuluu jo 10 sekuntia. Lähestymistapa ei tosin myöskään tällaisenaan tue likimääräistä sovittamista niin, että lievästi erilaiset toistuvat mallit luettaisiin samoiksi.

3.3 Audion sormenjäljen muodostaminen

Audion sormenjäljellä tarkoitetaan jokaiselle tietokannan kappaleelle erikseen muodostettua tunnistetta, jonka avulla kappale pyritään yksilöimään mahdollisimman hyvin. Sormenjäljen avulla voidaan myös säästää suuria määriä tilaa sekä nopeuttaa suoritettavaa hakua merkittävästi. Yleensä sormenjäljet muodostetaan äänen vaihtelua kuvaavista visuaalisista esityksistä eli spektrogrammeista muodostamalla kappaleesta niiden avulla erilaisia *tunnisteita* (hash), joista sormenjäljet rakentuvat. Tämän jälkeen näitä sormenjälkiä verrataan tietokannasta löytyvien kappaleiden sormenjälkiin, ja näistä vertailuista saadut sormenjälkiehdokkaat evaluoidaan myöhemmin vielä uudelleen oikean hakutuloksen aikaansaamiseksi [Wang 2003].

Sormenjälkien tuottamiseen on esitelty useitakin erilaisia tapoja [Chandrasekhar et al. 2011], mutta Wangin ehdottamalla spektrogrammin huippujen käytöllä on omat etunsa. Ensinnäkin spektrogrammien huiput säilyvät muuttumattomina huonoissa ja paljon taustamelua sisältävissä näytteissä, eivätkä ne katoa herkästi muiden piirteiden sekaan. Toiseksi spektrogrammien huiput noudattavat superpositioperiaatetta, eli vaikka spektrogrammi sisältäisi tietoa niin musiikista kuin taustamelustakin, niitä voidaan silti analysoida erillään. Muun muassa Shazam käyttää spektrogrammien huippuja sormenjälkien muodostamisessa.

Wangin [2003] mukaan vertailuissa käytettäville sormenjäljille on myös asetettava muutama ehto, jotka niiden kuuluisi täyttää. Ensinnäkin niiden täytyisi sisältää riittävän paljon entropiaa sekä olla sen verran vahvoja, että huonoistakin ääninäytteistä luodut tunnisteet saataisiin yhdistettyä tietokannan ”puhtaista” kappaleista tehtyihin tunnisteisiin. Tämän lisäksi jokainen sormenjäljen tunniste tulisi laskea vain tietyn lyhyen ajanjakson sisälle sijoittuvista pisteistä, jotteivät kaukaiset, tämän ajanjakson ulkopuolelle sijoittuvat tapahtumat, vaikuttaisi tunnisteen sisältöön. Tämä on tärkeää sen takia, että kappaleesta nauhoitettu näyte on yleensä paljon lyhyempi kuin kokonainen kappale, eikä täten voi sisältää tietoa kappaleessa paljon

aikaisemmin tai myöhemmin esiintyvistä tapahtumista. Siksi sormenjälkeen vaikuttavien tapahtumien täytyisi sijaita mahdollisimman lähellä sitä ajankohtaa, josta sormenjälki on tehty. Neljäs ehto on, että sormenjälkiä verratessa näytteen sormenjäljet tulee voida sijoittua aikajanalla mihin tahansa tietokannan näytettä. Tämä sen takia, että näytteet voivat olla nauhoitettu mistä kohtaa haettua kappaletta tahansa.

Monesti tunnisteet muodostuvat pareista, ja vaikka se saattaakin kasvattaa tilankäyttöä jopa kymmenkertaisesti, se samalla nopeuttaa hakuprosessia jopa 10000 kertaisesti.

Kun kappaleen kaikki sormenjäljet ja tunnisteet on saatu laskettua ja niitä kaikkia on haettu tietokannasta, jäljelle jää tulosten evaluoiminen. Yksinkertaisimmillaan saatuja ehdokkaita verrataan annettuun näytteeseen niin, että yhteensopivat piirteet esiintyvät samoin väliajoin suhteessa toisiinsa molemmissa kappaleissa. Vertailualgoritmien tarkemmat yksityiskohdat sivuutetaan tässä tutkielmassa (ks. [Wang 2003]).

3.4 MIDI, MP3 ja vektorit

Vektorit ovat yksi tapa tallentaa tietoa musiikista ja sen piirteistä. Esimerkiksi kromavektoreita voidaan käyttää, kun vertailuja halutaan tehdä MIDI- ja äänitiedostojen välillä, ja PC-vektoreita, kun vertailuja tehdään MP3-tiedostojen kanssa.

MIDI-tiedostot ovat 1980-luvulla kehitetty protokolla, jonka avulla elektroniset soittimet ja muut laitteet pystyvät kommunikoimaan keskenään. MIDI-tiedostojen etuihin lukeutuu muun muassa se, että ne ovat paljon kompaktimpi tapa säilöä musiikkia kuin varsinainen audiodata. Yksittäisen kappaleen tallentamiseen saatetaan tarvita vain pari sataa MIDI-viestiä, kun taas normaalia audiodataa ”sämplätään” jopa tuhansia kertoja sekunnissa. Tiedostojen kompakti koko perustuu siihen, että MIDI ei itsessään sisällä ääntä kuten MP3-tiedostot, vaan se koostuu sarjasta viestejä ja ohjeita, joiden avulla MIDI-instrumentit kuten syntetisaattorit pystyvät tuottamaan oikeanlaista ääntä. MIDI-tiedostojen toinen vahvuus on se, että niitä on myös helppo muokata ilman, että koko tiedostoa tarvitsee uudelleen nauhoittaa.

Yksinkertaisuutensa ja kompaktin kokonsa ansiosta MIDI-tiedostoja on toisinaan hyödynnetty myös musiikin analysoimisessa ja tunnistamisessa. Niitä voidaan hyödyntää musiikin tunnistamisessa vertailemalla MIDI-tiedostoista ja äänitiedostojen akustisesta datasta saatuja *kromagrammeja* [Hu et al. 2003]. Tällainen vertailu perustuu pelkästään eri sävelluokkien analysointiin ja monet muut piirteet voidaan jättää tässä tapauksessa kokonaan huomioimatta.

Aluksi akustinen data muutetaan kromagrammeiksi, eli kromavektoreiden joukoksi, jossa jokainen vektori koostuu 12 elementistä,

aivan kuin kromaattinen sävelasteikkokin *(C, C#, D, D#, E, F, F#, G, G#, A, A# B)*. Jokainen vektori kuvastaa tietyn, esimerkiksi 0,5 sekunnin, pituista kohtaa äänitiedostosta, ja jokainen vektorin elementti kuvastaa yksittäisen sävelen voimakkuutta juuri sillä hetkellä. Koska eri oktaavit on tässä toteutuksessa tiivistetty yhdeksi yleispäteväksi oktaaviksi, sävelet eivät enää hajaudu pitkin laajaa asteikkoa ja ne saadaan helpommin ja tehokkaammin luettua kroma-vektoreihin. Vektoreiden muodostamisen jälkeen vektorit vielä normalisoidaan analysoinnin helpottamiseksi ja suurten yksittäisten poikkeuksien vaikutuksen minimoimiseksi. Tämän jälkeen MIDI-tiedostolle muodostetaan omat kromavektorinsa samaan tapaan.

Kun äänitiedosto ja MIDI-tiedosto on saatu muutettua kromavektoreiksi, vektoreiden yhteensopivuutta voidaan tutkia muodostamalla *yhteensopivuus-matriisi* (similarity matrix) esimerkiksi euklidista etäisyyttä käyttämällä. Tämän jälkeen mahdolliset yhteensopivuudet haetaan etsimällä matriisiin muodostuneet reitit. Nämä reitit kuvastavat niitä kohtia, joissa vektoreiden väliset etäisyydet ovat olleet kaikista pienimpiä, eli toisin sanoen äänitiedostot muistuttavat eniten toisiaan.

MP3 eli MPEG-1 Audio Layer 3 on äänenpakkausmenetelmä, josta on vuosien saatossa noussut hallitseva tiedostomuoto äänen jakelussa. Se on myöskin hyvin kompakti tapa pakata digitaalista musiikkia, koska siinä leikataan pois muun muassa ääniä, joita ihmiskorva ei kuule helposti. MP3-tiedostot ovat kuitenkin paljon suurempia kuin MIDI-tiedostot, koska pelkkien ohjeiden sijaan MP3-tiedostot sisältävät oikeaa ääntä.

MP3-objektit jakautuvat ennalta määrätyn moneen saman pituiseen osaan, joita kutsutaan *jaksoiksi* (phase). Jaksot vastaavat pituudeltaan yleensä paremmin käyttäjän syöttämää näytettä kuin kokonaiset kappaleet, ja ne kuvastavat yleensä kappaleen yksittäistä lausetta. Jaksot taas koostuvat *katkelmista* (slot), jotka kuvastavat normaalisti yksittäistä säveltä tai sanaa laulussa. Kaikista pienimpiä MP3-tiedoston osia ovat *kehykset (*frame*)* ja *näytteet* (sample). Standardin mukaan kehys koostuu 1152:sta näytteestä ja äänisignaali voidaan jakaa 32:teen *alajoukkoon* (subband) [ISO/IEC 1 1172-3, 1993; Liun ja muiden [2001] mukaan].

PC-vektorit (polyphase-filter-bank coefficient) ovat 32-ulotteisia vektoreita, jotka kuvastavat äänisignaalin energian jakautumista jokaisen äänisignaalin alajoukon sisällä. Jos joissakin näistä 32 alajoukosta esiintyy selvästi muista poikkeavia äkillisiä ja suuria muutoksia, niiden ajatellaan olevan koko kyseisen MP3-jakson yksilöivimpiä ominaisuuksia.

Jokaisen aukon PC-vektori saadaan laskettua, kun sen sisällä olevien kehysten PC-vektoreille lasketaan keskiarvot. Näitä katkelmien PC-vektoreita kutsutaan paikallisiksi MP3-ominaisuuksiksi, ja näistä PC-vektoreista voi myös

muodostaa uusia vektoreita, jotka kuvastavat energian jakautumista jokaisen jakson sisällä. Näitä kehysten PC-vektoreita kutsutaankin globaaleiksi MP3-ominaisuuksiksi.

PC-vektorit kuvastavat yksittäisten jaksojen ja katkelmien piirteitä ja niiden avulla kappaleen jaksoja voidaan eritellä toisistaan vaikka ne kuulostaisivat samalta. Lopuksi jokaisen jakson PC-vektorista ja tämän jakson katkelmien PC-vektoreista muodostetaan uusi vektori, MFD-vektori eli *MP3:n ominaisuusvektori* (MP3 feature discriminator). Kahden kappaleen välisiä eroja on myöhemmin mahdollista vertailla asettamalla näiden MFD-vektorit matriisiin ja laskemalla näiden vektoreiden välisiä euklidisia etäisyyksiä. Tässä tutkielmassa tarkemmat yksityiskohdat sivuutetaan (ks. [Liu et al. 2001])

4. Kappaleiden järjestäminen tietokantaan

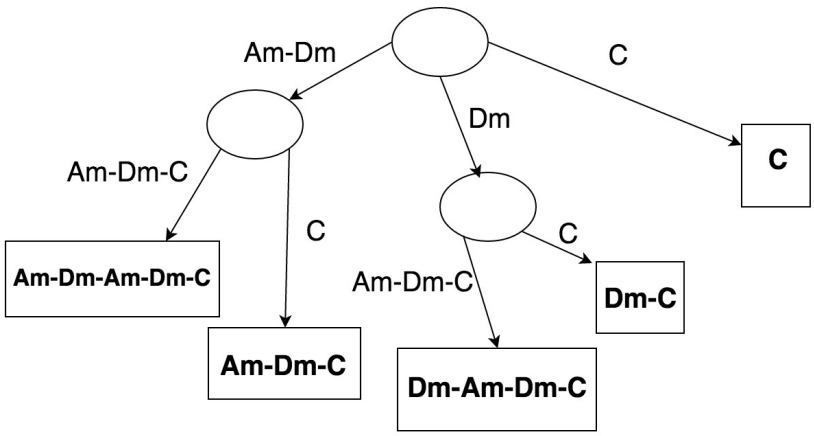
Yksi musiikin sisältöpohjaisen hakemisen perusongelmista on järjestää kaikki tietokannan alkiot niin, että sen sisällä olisi mahdollista suorittaa tarkkoja ja nopeita hakuja. Tämä tarkoittaa sitä, että tiedon löytymiseen tarvitaan mahdollisimman vähän alkioiden läpikäyntejä ja vertailuja, ja että hakutulokset saataisiin tuotettua niin nopeasti ja mahdollisimman pienellä virheasteella kuin mahdollista.

Musiikin hakemisen ja tunnistamisen yhteydessä tietokantojen varsinaiseen rakenteeseen ei monestikaan oteta kantaa, ja esimerkiksi Shazamin [Wang 2006] tietokannoista kerrotaan vain se, että hakuprosessi jakaantuu pääprosessorilta useammalle aliprosessorille, joista jokainen kattaa tietyn indeksin tietokantansa muistista. Nämä aliprosessorit lähettävät saadut hakutuloksensa pääprosessorille, joka suorittaa viimeiset valinnat jäljelle jääneiden ehdokkaiden välillä ja palauttaa tulokset käyttäjän nähtäväksi.

Yksi tapa indeksoida kappaleita tietokantaan on seurata Choun ja muiden [1996] esittämää tapaa käyttää tehtävään niin kutsuttuja *PAT-puita* (PAT-tree/Patricia tree/Suffix tree). Ne tukevat hakuja, joissa haku voidaan suorittaa mistä kohtaa kappaletta tahansa, mikä on välttämätöntä nopeamman hakutuloksen aikaansaamiseksi. PAT-puut muodostuvat tietokannassa olevien tekstinpätkien kaikista mahdollisista alimerkkijonoista, ja tässä kontekstissa tekstipätkillä tarkoitetaan lauluja ja alimerkkijonoilla pätkiä näistä lauluista. PAT-puita on mahdollista rakentaa sellaisten piirteiden avulla, jotka on mahdollista esittää merkkijonomuodossa. Choun ja muiden tutkimuksen esimerkissä tähän on käytetty sointuja ja niistä muodostuvia harmoniapätkiä.

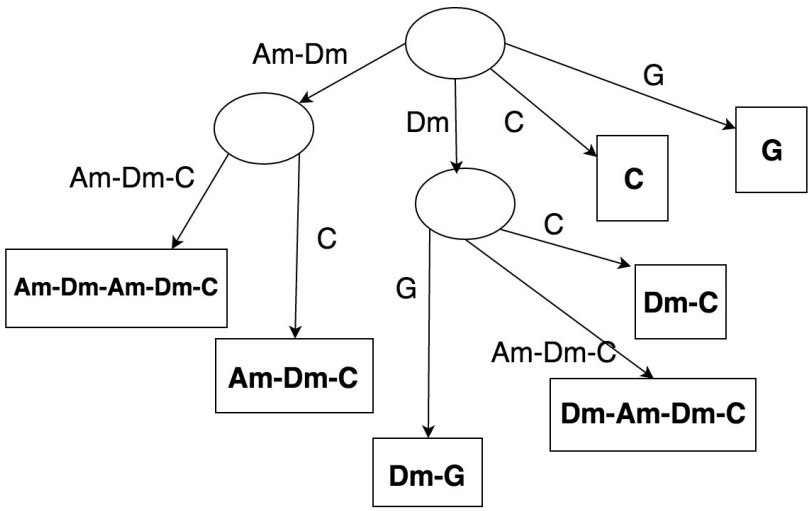
Kuvassa 3 oleva puu on muodostettu viidestä soinnusta koostuvalle merkkijonolle ”Am-Dm-Am-Dm-C”. Soikiolla merkityt solmut eli sisäsolmut tarkoittavat merkkijonon yksittäisiä merkkejä, joiden pituus voi olla enemmän kuin yksi. Neliöllä merkityillä solmuilla eli ulkosolmuilla tarkoitetaan kaikkia

”Am-Dm-Am-Dm-C”:n alimerkkijonoja. PAT-puuta luetaan lähtemällä juuresta liikkeelle, ja alaspäin edetään aina sitä mukaa, mitä haetusta merkkijonosta saadaan seuraavaksi luettua. Jos puusta halutaan löytää merkkijono ”Am-Dm-Am-Dm-C”, puussa siirrytään juuresta ensiksi vasempaan haaraan polkua ”Am-Dm” pitkin ja tämän jälkeen uudelleen vasemmalle pitkin polkua ”Am-Dm-C”. Näin päädytään ulkosolmuun ”Am-Dm-Am-Dm-C”, ja etsitty merkkijono on löydetty.



Kuva 3: PAT-puun yksinkertaistettu rakenne

Kuva 3 havainnollistaa hyvin myös PAT-puun tarjoamaa mahdollisuutta lähteä etsimään haluttua kohtaa keskeltä tekstiä tai kappaletta. Puurakennetta on myös helppo laajentaa lisäämällä sinne toisia kappaleita. Ensin lisättävä kappale muutetaan sointumuotoon aiemmin esitellyllä tavalla ja sitten soinnut jaetaan alimerkkijonoihin. Tämän jälkeen alimerkkijonot lisätään puuhun liikkumalla juuresta alaspäin niin kauan kun lisättävän merkkijonon etuliite löytyy puusta. Kun vastaavaa solmua ei enää löydy, luodaan sille uusi. Esimerkiksi kuvassa 4 puuhun on lisätty uusi merkkijono ”Dm-G”.



Kuva 4: Kuvan 3 PAT-puu, johon on lisätty merkkijono ”Dm-G”

5. Musiikin tunnistaminen ja kappaleiden vertailu toisiinsa

Musiikin piirteiden erottamisen jälkeen kappaleita täytyy vielä jotenkin vertailla toisiinsa, jotta etsitty kappale saataisiin haettua tietokannasta.

Tapahtumapohjaisella haulla (event-based search) tarkoitetaan hakua, jossa käsitellään yksittäisiä säveliä toisin kuin *kehyspohjaisessa haussa* (frame-based search), jossa käsitellään useasta sävelestä muodostettuja kehyksiä. Yksittäisiä säveltapahtumia käsitellessä ongelmaksi nousee yksittäisten sävelten tarkka ja vaivaton lukeminen. Tästä johtuen, vaikka kehyspohjainen haku onkin hakutapana hitaampi pidempien ja sitä myötä suurempien elementtien takia, on se silti useimmiten suositeltavampi. Kehyspohjaisessa haussa yksittäisten nuottien tarkan lukemisen ongelmaa ei nimittäin esiinny.

5.1 Likimääräistä sovittamista tukevat tekniikat

Varsinkin laulettuja ja hyräiltyjä näytteitä käytettäessä on tärkeää, että hakualgoritmit kykenevät joustavuuteen ja likimääräisten sovitusten suorittamiseen. Vain harvat ihmiset kykenevät tuottamaan kappaleesta täysin virheettömän näytteen, ja lähes aina haetun kappaleen ja käytetyn näytteen välillä on jonkin verran poikkeavuuksia. Tällaiset poikkeavuudet eivät kuitenkaan saa vääristää haun tuloksia, ja niinpä likimääräisen sovittamisen mahdollistamiseksi onkin kehitetty erilaisia tekniikoita.

Yksinkertaisimmillaan likimääräistä sovittamista voidaan tukea käyttämällä absoluuttisten arvojen sijaan niitä vastaavia suhteellisia arvoja. Toisin sanoen analysoinnissa ei välitetä niinkään esimerkiksi tarkoista sävelkorkeuksista tai kestoista, vaan niiden sijaan käytetään näiden arvojen

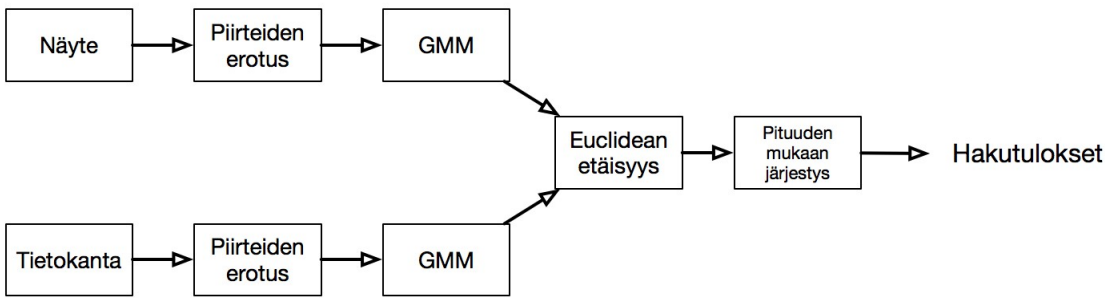
välisiä suhteellisia etäisyyksiä. Tällä tavalla saadaan ratkaistua muun muassa ongelma, joka ilmenee, kun hyräilty näyte hyräillään joko liian nopealla tai hitaalla temmolla.

Muutosetäisyyden (edit distance) [Hu and Dannenberg 2002] avulla kahden kappaleen samankaltaisuutta voidaan arvioida laskemalla, kuinka monta muutosta toiseen näytteistä täytyisi ensin tehdä, ennen kuin näytteet olisivat identtiset. Muutoksiksi luetaan muun muassa seuraavat viisi perustoimintoa: sävelen lisäys, poistaminen ja korvaaminen, sekä useamman sävelen sulauttaminen yhdeksi (yhdistely) ja yhden sävelen jakaminen useammaksi (osittelu). Muutosetäisyys ei itsessään kuitenkaan riitä, sillä se ei ota huomioon sitä, että vaikka kahden muutoksen etäisyydet olisivatkin yhtä korkeat, niin toinen muutoksista saattaa olla silti musiikillisesti toista tärkeämpi, ja näin ollen haun kannalta tärkeämpi [Liu *et al.* 1999].

Helén ja Virtanen [2007] esittävät tavan vertailla kahta kappaletta käyttämällä hyväksi euklidista etäisyyttä ja Gaussin sekomallia eli GMM:ää. Vaikka tutkimus keskittyykin pelkän musiikin sijaan kaikentyyppiseen ääneen, pätee se myös musiikin kohdalla.

Kuten kuvasta 5 näkyy, ensin tietokannan kaikille olioille sekä etsitylle näytteelle suoritetaan piirteiden erotus ja näille piirteille lasketaan GMM. Tämän jälkeen määritetään näiden väliset euklidiset etäisyydet ja saadut tulokset järjestetään etäisyyksien mukaan. Lopuksi saadut tulokset palautetaan käyttäjälle. Koska Helénin ja Virtasen menetelmä käsittelee kaikenlaisia ääniä, eikä pelkästään musiikkia, käyttäjä saattaa haluta saada hausta yhden tuloksen sijasta useita, esimerkiksi kaikki kymmenen ääntä, jotka parhaiten vastaavat annettua näytettä.

Euklidisen etäisyyden tilalle on ehdotettu myös muita vaihtoehtoja [Helén and Virtanen, 2009], kuten histogrammit tai Mahalanobiin etäisyys, mutta testien perusteella euklidinen etäisyys tarjosi parhaat tulokset.



Kuva 5. Helénin ja Virtasen [2007] ehdottaman vertailualgoritmin rakenne

On tietenkin ilmiselvää, että haun nopeuttamiseksi tietokannan olioiden piirteiden erotus sekä GMM:n laskeminen on suoritettava jo etukäteen ja tulokset tallennettava myöskin tietokantaan, josta ne ovat myöhemmin helposti luettavissa.

5.2 Merkkijonojen vertailualgoritmeista johdetut algoritmit

Koska musiikin temaattisia piirteitä voidaan käsitellä merkkijonomuodossa, voidaan musiikkia analysoitaessa käyttää hyödyksi useita tekstin hakuun kehiteltyjä algoritmeja, kuten Knuthin-Morrisin-Prattin algoritmia ja Boyerin-Mooren algoritmia, jotka ovatkin vuosien varrella kehittyneet hyvin nopeiksi ja tehokkaiksi hakualgoritmeiksi. Musiikin hakemiseen liittyy kuitenkin joitakin ongelmia, jotka eivät esiinny normaalissa merkkijonohaussa, ja sen takia hakualgoritmit eivät useinkaan sovellu musiikin hakemiseen suoraan sellaisenaan.

Yksi tapa hyödyntää tekstin hakua varten kehitettyjä algoritmeja on käyttää Liun ja muiden [1999] lähestymistapaa, jossa samanlaisten sävelten sijainneista merkkijonon sisällä luodaan linkitettyjä listoja (kuva 6). Jokainen listan alkio on muotoa (x;y), kuvaten tietokannan x:nnen olion y:ttä säveltä. Linkitettyjä listoja käyttämällä voidaan suorittaa joko sellaisia hakuja, jotka etsivät tietokannasta vain ne merkkijonot, jotka sisältävät haetun merkkijonon täysin sellaisenaan, tai vaihtoehtoisesti haussa voidaan mahdollistaa pieniä eroja merkkijonojen välillä, ja näin tukea likimääräistä sovittamista. Näitä erityyppisiä hakuja kutsutaan nimillä *täsmällinen merkkijono haku* (exact string matching) ja *likimääräinen merkkijono haku* (approximate string matching).

Olkoot A ja B kaksi tietokannasta löytyvää kappaletta, joiden melodioiden merkkijonot ovat:

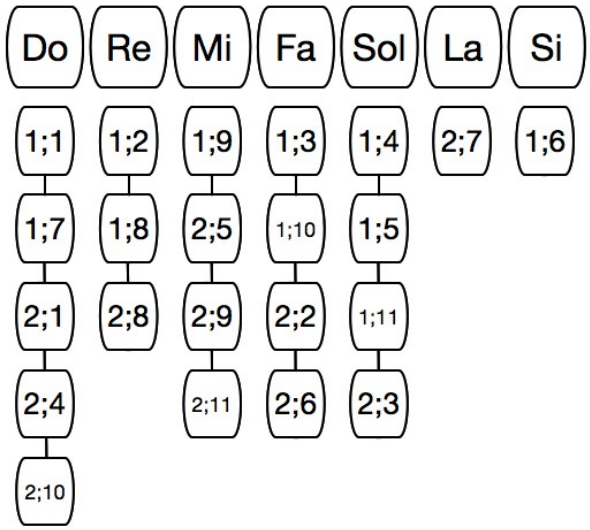
A = ”Do-re-fa-sol-sol-si-do-re-mi-fa-sol”

B = ”Do-fa-sol-do-mi-fa-la-re-mi-do-mi”.

Olkoon Q haussa käytetty näyte etsitystä kappaleesta. Q:n melodian merkkijono on seuraava:

Q = ”Do-mi-fa”.

Kuvassa 6 linkitetyt listat on muodostettu kaikille eri sävelille. Q:ta etsittäessä hakuun tarvitsee hakea kuitenkin vain listat, jotka käsittelevät sävelten ”do”, ”mi” ja ”fa” sijainteja. Tämän lisäksi listan alkuun ja loppuun luodaan apualkiot ”alku” ja ”loppu”, kuten kuvassa 7a. Tämän optimoinnin jälkeen haussa tarvitsee käydä läpi 22 sävelen sijaan enää 13.

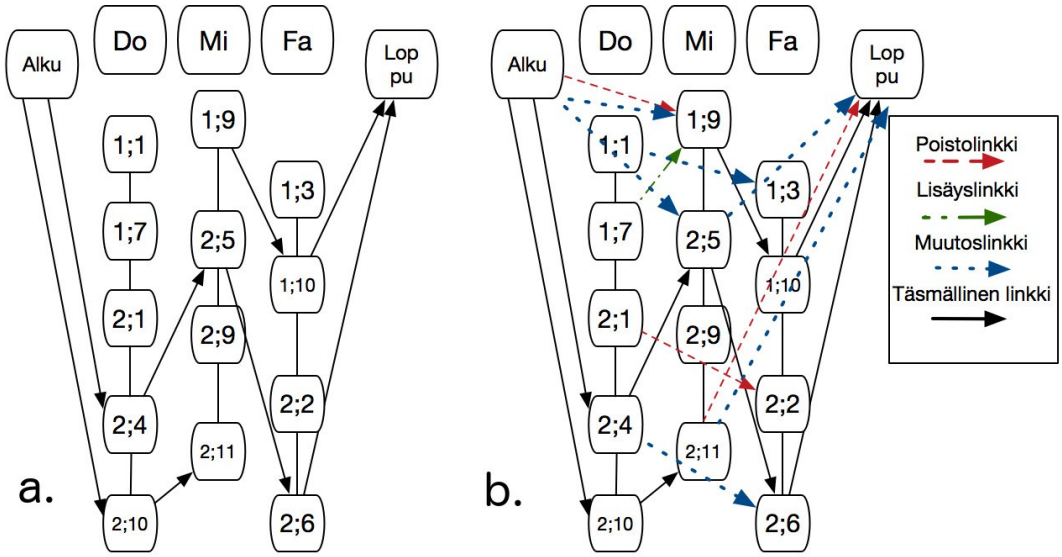


Kuva 6. Kaikista sävelistä muodostetut linkitetyt listat.

Merkkijonon Q varsinainen hakeminen aloitetaan luomalla täsmällisiä linkkejä jäljelle jääneiden linkitettyjen listojen välille. Toisin sanoen etsitään kaikki alimerkkijonot ”do-mi” ja ”mi-fa” ja yhdistetään nämä alimerkkijonot, jotta merkkijonon ”do-mi-fa” esiintymiset löydettäisiin tietokannasta.

Olkoon pos(ni) alkion ni sijainti merkkijonossa n. Täsmällisessä merkkijono haussa alkioiden n1 ja n2 välille luodaan täsmällinen linkki, jos pos(n1) + 1 = pos(n2), eli toisin sanoen n1-n2 on melodiamerkkijonon alimerkkijono. Täsmällinen linkki muodostetaan myös alku- ja loppualkioista ni:hin jos ni:llä on sisään tuleva tai ulos lähtevä täsmällinen linkki (kuva 7a). Täsmällinen haku on onnistunut, jos linkkien joukosta löytyy yksikin polku, joka lähtee alkualkiosta ja jatkuu aina loppualkioon saakka.

Likimääräistä merkkijono hakua käytettäessä hakuun lisätään vielä ominaisuus, joka sallii kolmenlaiset virheet: *poisto-* (dropout), *lisäys-* (insertion) sekä *muutosvirheet* (transposition errors). Esimerkiksi merkkijonot ”do-mi”, ”do-mi-mi-fa” ja ”do-?-fa” vastaavat kaikki haettua merkkijonoa Q muutosetäisyydellä 1. Linkitettyjen listojen väliin luodaan näiden muutosten perusteella täsmällisten linkkien lisäksi poisto-, lisäys-, ja muutoslinkkejä (kuva 7b). Lopuksi luodaan vielä linkit alku- ja loppualkioista kaikkiin niihin alkioihin jotka ovat joko polkunsa ensimmäisiä tai viimeisiä alkioita. Esimerkiksi kuvaan 7b lisättäisiin vielä linkit alkualkiosta alkioihin 1;1, 1;7 ja 2;1.



Kuva 7. Etsityn merkkijonon ympärille muodostetut linkit

Kun kaikki linkit on saatu luotua, kaikki polut käydään yksittäin läpi, ja haun tuloksiin luetaan jokainen polku, joka alkaa alkualkiosta ja jatkuu aina loppualkioon saakka. Muodostuneista linkitetyistä listoista voidaan nyt etsiä kaikki polut, jotka vastaavat haettua merkkijonoa joko täysin tai muutosetäisyydellä 1. Tämä lähestymistapa vaatii kuitenkin vielä jonkin verran hiomista, sillä tällaisenaan algoritmi sallii ainoastaan virheet, jotka poikkeavat haetusta muutosetäisyydellä 1.

Periaatteessa keinot hyödyntää tekstinhakualgoritmeja musiikin haussa ovat loputtomat, sillä heti kun musiikki saadaan muutettua merkkijonomuotoon, sitä voidaan käsitellä melko vapaasti tekstin tapaan.

6. Yhteenveto

Musiikin sisältöpohjaista hakemista varten on vuosien varrella kehitelty useita lähestymistapoja, jotka kaikki joko tehostavat aikaisempia algoritmeja, tai käsittelevät jotakin täysin uutta näkökulmaa. Tämä tutkielma on vain pintaraapaisu aiheeseen, eikä sen avulla pääse vielä tutustumaan algoritmeihin sen tarkemmin. Tutkielman tarkoitus on antaa lukijalle kattava perustietämys aiheesta ja auttaa häntä ymmärtämään miksi jotkin osa-alueet, kuten likimääräinen sovittaminen, ovat niin tärkeitä.

Musiikin tunnistusohjelmien kehittäjien on tärkeää ymmärtää, miten eri haku- ja näytetyypit vaikuttavat hakualgoritmien rakenteeseen ja mitkä musiikin piirteet ovat missäkin tilanteessa haun kannalta kaikista merkittävimpiä. Joissakin tilanteissa on tehtävä kompromissi haun nopeuden ja tarkkuuden välillä. Mitä nopeammin hakutulokset halutaan saada tuotettua,

sitä suuremmalla todennäköisyydellä virheellisten tulosten määrä kasvaa. Samalla tavalla jos tuloksien halutaan olevan mahdollisimman tarkkoja ja virheettömiä, sitä pidempään tulosten laskeminen ja analysoiminen tulee kestämään.

Eniten haasteita musiikin sisältöpohjaiseen hakuun tuovat QBH-haut, sillä niiden pitäisi sallia näytteessä suuria määriä virheitä, mutta silti samalla onnistua yhdistämään näyte yksittäiseen kappaleeseen. Vaikka QBH-hakuja varten on onnistuttu kehittämään monenlaisia joustavia algoritmeja, on niiden toimivuus yhä jokseenkin kyseenalainen, sillä usein käyttäjät eivät osaa vertailualgoritmien joustavuudesta huolimattakaan antaa ohjelmalle oikeanlaista näytettä, jonka avulla kappale saataisiin löydettyä. Tämän takia QBE-haut ovatkin paljon suositumpia tapa etsiä musiikkia sisältönsä perusteella.

Viiteluettelo

Chandrasekhar, V., Sharifi, M., Ross, D., A. (2011). Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications. In: *Proc. of the 12th International Society for Music Information Retrieval Conference,* 801-806.

Chou, T. C., Chen, A. L. P., Liu, C. C. (1996). Music databases: Indexing techniques and implementation. In: *Proc. of the 2nd International Workshop on Multimedia Database Management Systems,* 46-53.

Ghias, A., Logan, J., Chamberlin, D., Smith, B. C. (1995). Query by humming: musical information retrieval in an audio database. In: *Proc. of the 3rd ACM International Conference on Multimedia,* 231-236.

Helén, M., Virtanen, T. (2007). Query by example of audio signals using Euclidean distance between Gaussian mixture models. In: *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 225-228.

Helén, M., Virtanen, T. (2009). Audio query by example using similarity measures between probability density functions of features. *EURASIP Journal on Audio, Speech, and Music Processing.*

Hu, N., Dannenberg, R. B. (2002). A comparison of melodic database retrieval techniques using sung queries. In: *Proc. of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, 301-307.

Hu, N., Dannenberg, R. B., Tzanetakis, G. (2003). *Polyphonic Audio Matching and Alignment for Music Retrieval*. Department of Computer Science, School of Computer Science: 521.

Hsu, J. L., Liu, C. C., and Chen, A. L. P. (1998). Efficient repeating pattern finding in music databases. In: *Proc. of the 7th International Conference on Information and Knowledge Management*, 281-288.

ISO/IEC 1 1172-3. (1993). Information technology coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s.

Jones, G. T. (1974). *Music Theory*, Harper & Row.

Laaksonen, A. (2015). *Algorithms for Melody Search and Transcription.* Ph. D. Dissertation, Department of Computer Science, University of Helsinki.

Liu, C. C., Hsu, J. L., Chen, A. L. P. (1999). An approximate string matching algorithm for content-based music data retrieval. In: *Proc. of the 1st IEEE International Conference on Multimedia Computing and Systems,* 451-456.

Liu, C. C., Tsai, P. J. (2001). Content-based retrieval of MP3 music objects. In: *Proc. of the 10th International Conference on Information and Knowledge Management,* 506-511.

Thomas, V., Ewert, S., Clausen, M. (2012). Fast intra-collection audio matching. In: *Proc. of the 2nd International ACM Workshop on Music Information Retrieval with User-centered and Multimodal Strategies,* 1-6.

Wang, A. (2003). An industrial strength audio search algorithm. In: *Proc. of the 4th International Conference on Music Information Retrieval,* 7-13.

Wang, A. (2006). The Shazam music recognition service. *Communications of the ACM* 49, 8, 44-48.

Tekoelämä

Eetu Suonpää

Tiivistelmä.

Tekoelämä on tutkimusalueena melko iäkäs tietojenkäsittelyn osa-alue, josta ei kuitenkaan puhuta paljon. Joissain oppilaitoksissa tekoelämä tulee luonnostaan esiin tietojenkäsittelyopintojen edetessä, kun taas toisaalla asiasta ei välttämättä puhuta juuri ollenkaan. Tekoelämä on kuitenkin vielä nykyäänkin haastava ala, jolla on käyttökohteita esimerkiksi filosofiassa, tieteellisessä tutkimuksessa, algoritmiikassa ja jopa opetuksessa.

Tämän tutkielman tarkoituksena on antaa lukijalle sellainen perustietämys tekoelämästä, josta on helppo laajentaa omatoimisesti yhä syvempään tietämykseen aiheesta. Lähestyn tekoelämää nimenomaan tietojenkäsittelyn näkökulmasta ja esimerkiksi biologista tekoelämää ei käsitellä. Pyrin kertomaan tekoelämästä tutkimusalueena ja hyödyllisenä ongelmanratkaisun, opetuksen ja tutkimuksen välineenä. Lisäksi annan myös esimerkkejä kehitetyistä ohjelmistoista, joilla on mahdollista tutustua omatoimisesti tekoelämään ja siihen liittyviin tekniikoihin.

Aluksi tarkastelen jonkin verran tekoelämän laaja-alaista ja monipuolista historiaa aina ensimmäisistä itseään monistavien koneiden (Self-reproducing machine) ideoista nykyaikaiseen tekoelämätekniikoiden käyttöön. Esittelen muutamia eri määritelmiä ja jaotteluja, joita tekoelämälle voidaan antaa. Sen jälkeen esittelen kaksi tekoelämätekniikkaa, joilla tekoelämää voidaan hyödyntää mitä moninaisimmilla tavoilla ja esittelen myös muutamia sovelluksia, joilla tekoelämään ja siihen liittyviin tekniikoihin ja teorioihin tutustuminen on mahdollista. Lopuksi käyn läpi viimeaikaisia saavutuksia tekoelämän saralla sekä ongelmia, joita tekoelämän suhteen on vielä ratkaistavana.

Avainsanat ja -sanonnat: tekoelämä, soluautomaatit, toimijapohjainen simulaatio, itseään monistavat koneet, esiin nousevat mallit, filosofia, biologia.

1. Johdanto

Tekoelämä (Artificial Life, ALife) on tieteen ja tutkimuksen ala, jonka tarkoituksena on yrittää mallintaa sellaisia biologisia toimintoja, joista muodostuu meidän ymmärtämämme elämä. Ensimmäiset tekoelämään viittaavat ehdotukset teki John von Neumann 1940-luvulla, jonka jälkeen monet tutkijat ovat laajentaneet tekoelämän käsitettä. [McMullin 2000] Tekoelämä kattaa monia eri tieteenaloja varsin laaja-alaisesti ja sitä käytetäänkin monissa tutkimuksissa apuna, kun halutaan ymmärtää esimerkiksi jonkin laajemman kokonaisuuden toimintaa tai kehitystä. Käyttökohteina voi olla erilaisten mallien ja järjestelmi-

en opettaminen [Kopfler and Begel 2003], kollektiivisten makrotason toiminnallisuuksien muodostumisen ymmärtäminen [Ilachinski 2003] tai vaikkapa uuden algoritmin kehittäminen luonnonvalintaan perustuen [Gu et al. 2009].

Tekoelämään liittyen on vuosien saatossa kehitetty monia eri tekniikoita, kuten monimutkaiset mukautuvat järjestelmät, toimijapohjaiset simulaatiot, soluautomaatit ja neuroverkot. Tekniikoiden avulla on pyritty visualisoimaan, mallintamaan ja ymmärtämään erilaisia ilmiöitä, kuten yksittäisten toimijoiden muodostamasta joukosta esiin nousevia näennäisiä käyttäytymismalleja tai evoluutiota. Erityisesti nämä käyttäytymis- ja toimintamallit ovat tärkeässä osassa tekoelämän avulla tehtävää tutkimusta. Eräs tärkeä kysymys onkin, minkälaisia käyttäytymismalleja voidaan havaita, kun simuloidaan yksittäisten toimijoiden muodostamaa järjestelmää, jossa jokainen toimija reagoi toisiin toimijoihin ja toimii saamansa datan perusteella ilman keskinäistä ohjausta.

Koska tekoelämä on hyvin laaja-alainen tutkimusalue, jonka sisällä on kehitetty monia erilaisia tekniikoita ja teorioita, on tekoelämästä esitetty vuosien varrella useita erilaisia jaotteluja. Kenties yleisin tapa jaotella tekoelämää liittyy tekoelämän perusolemukseen, eli siihen, miten tarkasti tekoelämä voi mallintaa luonnollista elämää. Toinen, hyvin perustavanlaatuinen tapa, on jakaa tekoelämä sen perusteella, millä alalla ja mitä menetelmiä käytetään. Vaikka käytetyt teoriat ovat samoja, biologien ja insinöörien käyttämät menetelmät saattavat erota toisistaan hyvinkin paljon [Xuyan 2005].

Tietojenkäsittelytieteen kannalta ehkä tärkeimmät saavutukset tekoelämän tutkimuksen ja elämän mallintamisen saralla ovat olleet erilaisten tekoelämäsovellusten kehittäminen. Game of Life -sääntöjä käyttäviä ohjelmistoja, joilla on mahdollista mallintaa ja tutkia elämää soluautomaatin avulla, on ollut olemassa jo kauan. Myös sovelluksia, jotka käsittelevät yksittäistä mallia, kuten vuorovesi-ilmiötä, evoluutiota tai genetiikkaa, on ollut jo kauan. Sen sijaan uudempi ja yleisesti ottaen ehkä tärkeämpi kehitys on tapahtunut niin sanottujen yleisten tekoelämäsovellusten saralla. Monet yksityiset tutkijat ja tutkimuslaitokset ovat kehittäneet sovelluksia, joiden avulla voidaan luoda, tutkia ja opettaa periaatteessa mitä tahansa malleja. Nämä sovellukset sisältävät vain vähän ennalta määrättyjä sääntöjä ja niiden tarkoituksena onkin antaa käyttäjälle mahdollisuus luoda itse omat sääntönsä. Sovellukset tarjoavat ympäristön, jonka sisällä voidaan rakentaa eri tekoelämätekniikoita käyttäen mitä ihmeellisimpiä maailmoja ja olioita. Kun käyttäjä on määritellyt haluamansa säännöt ja toimijat, voi hän ajaa simulaation ja tutkia, miten eri toimijat käyttäytyvät määrättyjen sääntöjen perusteella. Näiden sovellusten avulla käytännössä kuka tahansa voi helposti alkaa tutkia tekoelämän tarjoamia mahdollisuuksia, ja selvästi suurin hyöty niistä tuleekin opetuksen alalla.

Vaikka tekoelämästä onkin paljon hyötyä monella eri alalla, ei se kuitenkaan ole täydellinen ratkaisu kaikkiin ongelmiin. Ehkä selvin ongelma ilmenee, kun tekoelämää käytetään tieteellisessä tutkimuksessa. Erityisesti ihmisen evoluution tutkimuksessa ongelmaksi muodostuu tekoelämän mekaanisuus. Tutkijat ovat väitelleet, voiko tekoelämän avulla tutkittu evoluutio tuottaa realistisia ja luotettavia tuloksia, kun yksittäiset toimijat ovat epäinhimillisiä tietokonesimulaatioita. Tieteellinen tutkimus perustuu luotettaviin, todistettaviin ja toistettavissa oleviin tuloksiin, joita tekoelämän avulla ei vielä tähän mennessä olla saatu kunnolla aikaiseksi. Tekoelämän tuloksia voidaan tutkia analyyttisesti ja tuloksista voidaan johtaa teorioita esimerkiksi käyttäytymismallien syntymiselle, mutta saadut tulokset ovat usein melko subjektiivisia ja epätieteellisiä. Toinen, ehkä melko itsestäänselvä ongelma tekoelämässä on kysymys, mitä on elämä ja voidaanko tekoelämää pitää elämänä? Monet kokevat tekoelämän ongelmalliseksi juuri siitä syystä, että tekoelämäsimulaatiot ajetaan tietokoneen sisällä, eikä tekoelämällä silloin ole mitään tekemistä biologisen, niin sanotusti ”aidon” elämän kanssa. Vastaväitöksenä tälle on esitetty, että elämä ei suinkaan ole pelkästään biologinen tapahtuma, vaan tiettyjen sääntöjen mukaan tapahtuva prosessi, jota voidaan mallintaa tietokoneella.

Tutkielman toisessa luvussa esittelen tekoelämän historiaa aina ensimmäisistä teorioista nykyaikaan. Kolmannessa luvussa esitän erään määritelmän tekoelämälle, sekä lisäksi muutaman tavan jakaa tekoelämän käsitettä. Luvussa 4 esittelen kaksi erilaista, mutta hyvin olennaista tekniikkaa, joilla tekoelämää voidaan toteuttaa. Sen jälkeen luvuissa 5 ja 6 käyn läpi tekoelämään liittyviä ongelmia, sekä siitä saatavia hyötyjä. Lopuksi, luvussa 7, käyn lyhyesti läpi tutkielman sisällön.

2. Tekoelämän historiaa

Tekoelämän syntykohdaksi ei voida sanoa yhtään tiettyä pistettä historiassa, mutta ensimmäiset tekoelämään liittyvät tekstit ilmestyivät 1940-luvun lopulla John von Neumannin julkaisemina artikkeleina [McMullin 2000]. Von Neumann esitti artikkeleissaan teoreettisen itseään monistavan (monimutkaisen) koneen, joka koostuu kaksiulotteisesta, 20 000 solua käsittävästä taulukosta. Jokaisella näistä soluista voi olla 29 tilaa, jotka muuttuvat tiettyjen sääntöjen perusteella. Tässä järjestelmässä on mahdollista saavuttaa tilanne, jossa tietyssä tilassa olevien solujen muodostama joukko monistuu tietyn ajan kuluttua. Tällaista konetta tai järjestelmää, jossa jokin joukko voi monistaa itsensä, kutsutaan itseään monistavaksi järjestelmäksi. Tämä oli yksi ensimmäisistä teorioista, joiden voidaan lukea kuuluvan nykyisenlaisen tekoelämän piiriin.

Von Neumannin järjestelmä koostui hyvin suuresta määrästä pieniä, toisiinsa vaikuttavia osia ja triviaaleja sääntöjä, eikä von Neumann koskaan määritellyt formaaleja sääntöjä monimutkaisuudelle. Tästä johtuen yksinkertaisin itseään monistava järjestelmä voidaan muodostaa säännöllä, jonka mukaan jokainen ”elossa” oleva solu vaikuttaa naapureihinsa, saaden naapurisolut ”eloon”. Monet tutkijat esittivät kritiikkiä von Neumannin tutkimuksen merkittävyydestä ja ovat ajan saatossa yrittäneet eri tavoin määritellä monimutkaisuutta ja sitä, mikä on todellisuudessa pienin itseään monistava monimutkainen järjestelmä. 1900-luvun loppupuolella mm. C.G. Langton, Arthur Burks ja John Conway yrittivät antaa omat määritelmänsä monimutkaisuudelle sekä kehittivät omia versioitaan yksinkertaisemmasta ratkaisusta. Burks ehdotti 1970, että monimutkaisen koneen olisi kyettävä peruslaskutoimituksiin. Langton sen sijaan ehdotti vuonna 1984, että soluautomaattien ja tekoelämän tulisi keskittyä nimenomaan heijastamaan luonnollisen elämän ominaisuuksia. Hänen mukaansa itseään monistavien koneiden tulisi koostua perimästä (genotype) ja ulkoasusta (phenotype). Näin ollen myös evoluutio olisi mahdollista itseään monistavien järjestelmien eri sukupolvien välillä. Tekoelämän kehitys onkin yleisesti ottaen seurannut Langtonin ehdotusta ja useat eri mallit ja simulaatiot perustuvat ehdotukseen perimästä ja ulkoasusta. [McMullin 2000]

Vaikka Langtonin ehdotus menestyi hyvin ja tutkijat pitävät sitä edelleen tärkeänä ja perustavanlaatuisena teoriana, myös Conwayn teoria on kasvattanut suosiotaan vuosien saatossa. Conway pyrki yksinkertaistamaan von Neumannin teoriaa suuresti. Pyrkimyksen tuloksena oli paljon von Neumannin soluautomaattia yksinkertaisempi järjestelmä nimeltä Game of Life. Conwayn tavoitteena oli luoda mahdollisimman yksinkertainen sääntöjoukko, jonka pohjalta soluautomaatilla voitaisiin luoda hyvinkin monimutkaisia järjestelmiä. Vaikkei Game of Life välttämättä korvannut tai edes yksinkertaistanut von Neumannin teoriaa, nousi se kuitenkin myös tavallisen kansan tuntemaksi teoriaksi. Game of Life on vielä tänäkin päivänä monien tietojenkäsittelystä kiinnostuneiden ensimmäisiä ohjelmointiprojekteja, ja teoriaan perustuu monia erilaisia soluautomaattisovelluksia [Dekker 1994].

Viime aikoina tekoelämä ei ole kehittynyt juurikaan teorioiden puolesta, mutta paljon kehitystä on tapahtunut siinä, miten tekoelämää ja sen eri tekniikoita voidaan hyödyntää ja soveltaa, ja tekniikoiden määrä on lisääntynyt valtavasti. Eri alojen tutkijat voivat yhdistää tutkimuksessaan vaikkapa soluautomaattien perusrakennetta neuroverkkojen tai geneettisten algoritmien kanssa. Tästä yksittäisten ominaisuuksien yhdistelemisestä johtuen tekoelämän kehitys näyttäisi siirtyneen pois vanhoista menetelmistä. Esimerkiksi evoluution tutkimus ei välttämättä ole kovin mielekästä käyttäen pelkästään soluautomaattien

tarjoamia mahdollisuuksia, mutta yhdistettynä geneettisiin algoritmeihin voidaan ongelmaa tarkastella realistisemmalla tavalla. Vaikka vanhoille tekniikoille ja teorioille on edelleen paikkansa, tekoelämän tulevaisuus vaikuttaa suosivan näiden yhdistelemistä ongelmakohtaisesti, ja lisäämällä tutkimuksen vaatimat asiat.

3. Määritelmä ja osa-alueita

3.1 Määritelmä

Jotta tekoelämän määritelmän käsittely olisi mielekästä, on hyvä ensin hieman pohtia luonnollisen elämän määritelmää. Luonnollisen elämän määrittely ei kuitenkaan ole kovin yksinkertainen tehtävä. Jokainen meistä tietää ja tunnistaa elämän, kun sellaista näkee, jokainen meistä tietää olevansa elossa, mutta tästä huolimatta elämän määritteleminen ei ole millään tavalla triviaali tehtävä. Xuyan [2005] määrittelee luonnollisen elämän biologian jatkuvana prosessina. Tarkemmin sanottuna, luonnollisella elämällä on muutamia ominaisuuksia: luonnollinen kasvu ja kehittyminen, joka ei ole ihmisen aikaansaamaa, lisääntyminen, niin kasvien, eläinten kuin ihmistenkin keskuudessa, luonnollisen elämän perustuminen orgaaniseen aineeseen sekä olion sisäiset toiminnot, ulkoinen käyttäytyminen ja mahdollinen älykkyys ja tunteet. Määritelmä on kuitenkin melko yleinen, eikä välttämättä määrittele luonnollista elämää riittävän tarkasti, jotta siitä olisi hyötyä. Xuyan itsekin toteaa tämän kaltaisen määritelmän sisältävän monia ongelmia, kuten mistä elämä on lähtöisin, mitä ovat tunteet ja minkälaiset oliot voivat tuntea ja mitä on älykkyys.

Sen sijaan esimerkiksi Coles [2009] ei anna valmista määritelmää, vaan tyytyy toteamaan, että elämän määritteleminen on vaikeaa. Ongelmaksi elämän määritelmässä vaikuttaa tulevan juuri luonnollisen elämän monipuolisuus. Coles antaa esimerkkinä kynttilän liekin; se täyttää luonnollisen elämän määritelmän (biologinen toiminta happea polttamalla, sisältää ainetta, pystyy lisääntymään..), mutta varmastikaan moni ei väittäisi liekin olevan elossa.

Tekoelämään käsitettä on myös yritetty vuosien varrella määritellä, mutta vielä tähän päivään mennessä sille ei ole kehittynyt yhtään selkää, yksiselitteistä määritelmää. Tämä johtunee siitä faktasta, että jo luonnollisen elämän määritteleminen on vaativa, lähes mahdoton tehtävä. Mutta samoin kuin luonnolliselle elämälle, myös tekoelämälle voidaan antaa erilaisia ominaisuuksia, ja se voidaan myös tietyiltä osin määritellä näiden ominaisuuksien perusteella. Eräs yleisimmistä määritelmistä tekoelämälle saatiin vuonna 1987, kun Langton esitti ajatuksiaan ja teorioitaan tekoelämästä. Langtonin mukaan tekoelämä tarkoit-

taa sellaisen ihmisen luoman järjestelmän tutkimusta, jossa esiintyy elämälle tyypillisiä ominaisuuksia, kuten lisääntymistä ja sen myötä evoluutiota. Lisäksi tekoelämä tarjoaa mahdollisuuden simuloida ja tutkia luonnollista elämää ja sen olioita, sekä mahdollistaa uudenlaisen näkökulman teoreettiseen biologiaan; voidaan pohtia, millaista elämä *voisi* olla, sen sijaan että tutkitaan, millaista elämä *on*. [Xuyan 2005]

3.2 Osa-alueita ja jaottelua

Tekoelämätutkimus kattaa monia eri aloja ja tekniikoita. Näin ollen eri alojen tutkijat saattavat käyttää hyvinkin toisistaan poikkeavia menetelmiä hyödyntäessään tekoelämää. Toinen hyvin selvä ja perustavanlaatuinen tapa jaotella tekoelämää liittyy vahvasti yksittäisten ihmisten omaan subjektiiviseen näkemykseen tekoelämästä. Tästä johtuen tekoelämää ei ole yksimielisesti jaoteltu selviin osa-alueisiin, vaikka tietynlaiset jaottelut ovatkin yleisesti hyväksyttyjä ja käytettyjä.

Eräs ehdotus tekoelämän jaotteluksi perustuu käytettyihin menetelmiin. Xuyan [2005] ehdotti, että Langtonin määritelmää voitaisiin edelleen kehittää, jotta se kuvastaisi paremmin tekoelämän suhdetta luonnolliseen elämään. Langtonin määritelmä käsittelee tekoelämää melko suppealta kannalta, eikä se esimerkiksi sisällä mitään yhtymäkohtaa biologiaan. Xuyan ehdotti kokonaan uutta termiä yleistetty tekoelämä (Generalized Artificial Life, GAL). Tämä yleisen tekoelämän määritelmä ei itsessään ota kantaa siihen, mitä tekniikoita tai materiaaleja käytetään, vaan pyrkii yleistämään tekoelämän perusidean. Yleistetty tekoelämä tarkoittaa siis ihmisen luomaa elämää, jolla on luonnollisen elämän ominaisuuksia ja joka tarjoaa mahdollisuuden mallintaa ja jatkaa luonnollista elämää. [Xuyan 2005]

Xuyan esittää myös kolme eräänlaista alakategoriaa yleistetylle tekoelämälle, jotka ottavat enemmän kantaa käytettyihin menetelmiin ja materiaaleihin. Alakategoriat ovat biologinen tekoelämä (Biological Artificial Life, BAL), rakennettu tekoelämä (Engineering Artificial Life, EAL) ja näitä kahta yhdistävä bio-rakennettu tekoelämä (Bio-Engineering Artificial Life, BEAL). Näiden tarkoituksena on eritellä esimerkiksi biologinen tekoelämä niin sanotusta rakennetusta, tai ”suppeasta” tekoelämästä. Esimerkkinä biologisesta tekoelämästä on kloonattu lammas, kun taas rakennettu tekoelämä käsittelee pelkästään ei-biologisia asioita. Näiden yhdistelmä, bio-rakennettu tekoelämä, on Xuyanin määritelmän mukaan kaikista lähimpänä luonnollista, eli ”oikeaa” elämää. [Xuyan 2005]

Toinen tapa jaotella tekoelämää on enemmänkin riippuvainen jaottelun tekijästä ja sen myötä usein melko subjektiivinen. Tekoelämä määritellään usein ihmisen luomana elämänä, oli se sitten biologista (kuten kloonaaminen) tai digitaalista. Määritelmästä herää kuitenkin heti kysymys: ”onko tekoelämä oikeaa elämää?”. Koska jo itsessään elämän määritteleminen on vaikeaa, ellei jopa mahdotonta, on tekoelämän ontologinen asema edelleen kiistanalainen aihe. Joidenkin ihmisten mielestä elämän yhdistäminen tietokonesimulaatioon on samanlainen virhe kuin vaikkapa elämän yhdistäminen tauluun. Vastaavasti toiset ihmiset kokevat, että tietokonesimulaatiot, joissa esiintyy esiin nousevia malleja (kuten parvikäyttäytyminen), eivät edusta niinkään malleja, vaan ovat mallien yksittäisiä esiintymisiä. Tämä ajattelutapa simulaatioista tiettyjen mallien esiintyminä on nimeltään vahvan tekoelämän kanta (Strong ALife Position). [Coles 2009]

Vaikka tekoelämälle on esitetty jaottelutapoja ja osa-alueita, on se siitä huolimatta edelleen hyvin vaikeasti jaoteltavissa. Osittain syynä on sen laaja-alaisuus ja toisaalta taas elämän ja tekoelämän määriteltävyyden vaikeus. Lisäksi ongelmana on, miten vahvasti tekoelämä voi, tai edes pystyy, mallintamaan ”oikeaa” elämää. Vaikka jaottelua on tehty, ei se ole välttämättä levinnyt yleiseen käyttöön.

4. Tekoelämän tekniikoita

4.1 Soluautomaatit

Tekoelämän ja sen tutkimuksen alalla on useita kymmeniä erilaisia malleja, kuten erilaisia yksittäisiin toimijoihin perustuvia simulaatiomalleja, sekä neuroverkkoja ja genetiikkaa yhdistäviä malleja. Siitä huolimatta eräs vanhimmista, mutta edelleen käytetyimmistä malleista on soluautomaatit (cellular automata, CA). Monet kehittyneemmät mallit, kuten vaikkapa edellä mainitut yksittäisten toimijoiden simulaatiot, pohjautuvat yhä vahvasti soluautomaatteihin ja sisältävät soluautomaateille tyypillisiä ominaisuuksia. Soluautomaatteja on käytetty laajasti jo 1940-luvun lopulta asti, jolloin von Neumann kehitti ensimmäiset tekoelämään liittyvät teoriat monimutkaisesta itseään kehittävästä koneesta, käyttäen nimenomaan soluautomaattia. Von Neumannin jälkeen tekoelämä ja sen teoriat ovat kehittyneet huimasti ja luoneet uusia teorioita, mutta soluautomaatit ovat edelleen eräänlaisena kivijalkana tekoelämän tutkimuksessa.

Soluautomaateilla tarkoitetaan tyypillisesti kaksiulotteista taulukkoa, tai solukkoa, jonka jokaisella alkiolla (ruudulla, solulla) on joku tila, kuten *elossa* tai *kuollut*. Nämä solut reagoivat solukossa olevien muiden solujen kanssa ja nii-

den sisäinen tila muuttuu tiettyjen sääntöjen perusteella. Esimerkkinä vaikkapa solu, joka on aluksi *elossa*, mutta mikäli sen naapurisolu vaihtuu myös tilaan *elossa,* kyseinen solu kuolee. Tämän kaltaisten sääntöjen ja ennalta määriteltyjen tilojen perusteella voidaan luoda hyvin laaja joukko erilaisia niin sanottuja tilakoneita, jotka ovat jokaisella ajan hetkellä t jossakin tietyssä tilassa. Vaikka soluautomaatit ovat usein kaksiulotteisia, on historian saatossa kehitetty niin yksiulotteisia, elementaarisia soluautomaatteja (elemantary cellular automata) kuin moniulotteisia soluautomaatteja. Nykyään monet tutkijat suosivat kolmiulotteisia soluautomaatteja, sillä kasvaneen prosessointitehon avulla voidaan mallintaa hyvinkin laajoja ja monimutkaisia koneita niin sanotusti aidommassa tilassa kuin kaksiulotteisilla automaateilla.

Eräs tärkeä tekijä soluautomaattien, ja yleisesti myös tekoelämän, tutkimuksessa ja kehityksessä oli Langton. Sen lisäksi, että hän ensimmäisten joukossa käsitteli tekoelämää tutkimusalana, hän myös tutki kattavasti soluautomaattien mahdollisuuksia tekoelämän suhteen. Vuonna 1986 julkaisemassaan artikkelissa Langton vertaa yksityiskohtaisesti tekoelämän suhdetta formaalimpiin dynaamisiin järjestelmiin sekä pyrkii esimerkkien avulla demonstroimaan mahdollisuuden luoda tekoelämää soluautomaatteja käyttäen. Langton [1986] painottaa sitä, miten luonnollinen elämä koostuu pienistä yksittäisistä toimijoista, joiden kanssakäymisen tuloksena syntyy ilmiöitä. Näiden ilmiöiden joukko voidaan käsittää elämäksi. Artikkelissa Langton tukeutuu vahvasti soluautomaatteihin, sillä niiden avulla voidaan melko helposti muodostaa järjestelmä, josta nousee esiin monimutkaisia ilmiöitä ja malleja. [Langton 1986]

Soluautomaateilla rakennettavat tilakoneet muodostuvat yleensä joukosta sääntöjä, soluille määritellyistä mahdollisista eri tiloista, kuten *elossa* tai *kuollut*, sekä jonkinlaisesta solujen alkumuodostelmasta. Tämä alkumuodostelma voi olla satunnainen tai tarkoin harkittu, riippuen tutkittavasta ilmiöstä tai muodostettavasta tilakoneesta. Huomionarvoinen asia soluautomaateista on, että jopa hyvin yksinkertaisilla säännöillä, solujen tiloilla ja pienillä solukoilla, voidaan saavuttaa hyvinkin monimutkaisia järjestelmiä. Esimerkkinä vaikkapa Turingin koneet, jotka voivat suorittaa mitä tahansa laskutoimituksia, toteutettuna Game of Life -soluautomaatissa, josta myöhemmin lisää.

Soluautomaattien käyttökohteet ovat jotakuinkin rajattomat. Soluautomaateilla voidaan tutkia, todistaa, kehittää tai vaikkapa opettaa mitä erilaisimpia asioita. Yksi tärkeä tutkimuksen kohde ovat erilaiset satunnaisesta joukosta esiin nousevat mallit ja ilmiöt, esimerkiksi parvikäyttäytyminen tai monimutkaiset sosiaaliset kanssakäymiset. Koska soluautomaatti on lähinnä teoreettinen järjestelmä, voidaan sen soluille määritellä mielivaltaisesti haluttu määrä eri tiloja sekä solujen toimintasäännöt. Myös käytettävän solukon koon voi valita

tutkittavan asian mukaan. Näin ollen edes soluautomaattien yleensä kaksiulotteinen luonne ei ole aivan välittömänä esteenä monimutkaisia malleja tai teorioita tutkiessa. Mikäli esimerkiksi halutaan mallintaa erilaisten kaasujen toimintaa erilaisissa ympäristöissä, voidaan yhden solun valita esittävän yksittäistä kaasuhiukkasta ja säännöksi asettaa haluttu yhtälö.

Koska soluautomaatit toimivat tiettyjen, yksittäisille soluille sovellettavien sääntöjen perusteella, on soluautomaattien historiassa jokusia merkittäviä säännöstöjä. Näitä säännöstöjä käytetään ja tutkitaan edelleen ja ne tarjoavat hyvän lähtökohdan soluautomaattien, ja sen myötä tekoelämän, opettelemiseen. Erään vanhimmista ja merkittävimmistä soluautomaattisäännöstöistä, Game of Lifen, kehitti Conway vuonna 1970. Conwayn tavoitteena oli radikaalisti yksinkertaistaa von Neumannin teoreettista itseään monistavaa konetta, ja sen myötä koko von Neumannin soluautomaattia. Tuloksena oli yksi tunnetuimmista soluautomaateista, jolla on edelleen paljon hyötyä, erityisesti tietojenkäsittelyn alalla. Game of Lifen perustana oli ajatus siitä, että yksittäisten solujen tulisi olla mahdollisimman yksinkertaisia, eli niiden välisten toimintasääntöjen tulisi olla yksinkertaisia. Myöskin solujen mahdollisia tiloja tuli rajoittaa vain muutamaan. Näin ollen voitaisiin helposti tutkia esiin nousevia ilmiöitä, kuten solujen keskinäisiä reaktioita, muurahaisyhdyskunnan näennäistä monimutkaisuutta tai vaikkapa parvikäyttäytymistä. Game of Life toimii rajattoman suuressa kaksiulotteisessa taulukossa, jossa jokainen sen solu voi olla joko *elossa* tai *kuollut*. Jokainen näistä soluista toimii vain kahdeksan lähimmän naapurinsa kanssa seuraavien kolmen säännön perusteella [Alfonseca 1999]:

1) Jokainen *kuollut* solu herää henkiin, mikäli solulla on tasan kolme *elossa* olevaa naapuria (periytyminen).

2) Jokainen *elossa* oleva solu, jolla on vähemmän kuin kaksi tai enemmän kuin kolme *elossa* olevaa naapuria, kuolee (ali- tai ylikansoitus).

3) Kaikissa muissa tapauksissa, solu säilyttää edellisen tilansa.

Eri versioita säännöistä on Conwayn ensimmäisen ehdotuksen jälkeen tullut lukemattomia, mutta edellä mainittu säännöstö on edelleen niin sanotusti ”aito oikea”. Game of Lifen alkuasetelma on lähtökohtaisesti satunnainen (satunnainen määrä satunnaisia soluja on *elossa*), mutta esimerkiksi useat Game of Life -sovellukset tarjoavat käyttäjälle mahdollisuuden valita alkuasetelma mielivaltaisesti. Mahdollistamalla manuaalinen alkuasetelman luominen voidaan tutkia, mitä erilaisia järjestelmiä ja malleja Game of Life -soluautomaatista voi nousta esiin. Eräs maininnan arvoinen (ja myös pienin tähän asti löydetty) esiin nouseva malli on niin sanottu Liitäjä (glider), jota on hyödynnetty monien muiden mallien löytämiseen. Game of Life on myös todistettu laskennallisesti

täydelliseksi soluautomaatiksi (kykenee laskemaan minkä tahansa laskentatehtävän, samalla tavalla kuin Turingin kone) [Alfonseca 1999].

4.2. Toimijapohjainen simulaatio

Soluautomaatit ovat tärkeässä roolissa monilla eri tieteen ja tutkimuksen aloilla. Niiden avulla voidaan rakentaa hyvinkin monimutkaisia järjestelmiä, jotka kykenevät monistamaan itseään ja suorittamaan käytännössä mitä tahansa laskuja. Soluautomaattien perusrakenne koostuu yksinkertaisista toimijoista, soluista, jotka reagoivat tiettyjen sääntöjen perusteella ympäristönsä (esimerkiksi naapurisolujen) kanssa. Vaikka soluautomaateilla pystytään mallintamaan monia erilaisia malleja ja tutkimaan niistä esiin nousevia ilmiöitä ja käyttäytymismalleja, ovat ne nykyään monesti hieman riittämättömiä. Soluautomaattien pohjalta on kehittynyt monia, usein hieman monimutkaisempia malleja, jotka vastaavat paremmin eri alojen vaatimuksiin. Yksi tällainen malli on toimijapohjainen simulaatio (Agent-Based Simulation, ABS), jolla on monimutkaisuudestaan huolimatta paljon yhtäläisyyttä soluautomaatteihin.

Toimijapohjaisen simulaation määritelmä on hyvin häilyvä, sillä eri tutkimusalat määrittelevät sen usein hyvin eri tavalla. Sen lisäksi myös tutkimusalojen sisällä voidaan tilanteesta riippuen käyttää hieman erilaisia määritelmiä, tai jopa kokonaan eri nimeä. Kuitenkin määritelmä, kuten koko malli, pohjautuu suurelta osin soluautomaattien määritelmään. Toimijapohjainen simulaatiomalli perustuu yksittäisiin älykkäisiin toimijoihin (agent), jotka voivat liikkua, kommunikoida ja oppia asioita yhdessä muiden toimijoiden kanssa (vrt. soluautomaatit, joissa yksittäiset solut pysyvät paikallaan, mutta vaihtavat tilaansa naapurisolujensa tilojen perusteella). Tila, jossa simulaatio voidaan suorittaa, on käytännössä riippuvainen pelkästään simulaation luojasta. Simulaatio voidaan suorittaa vaikkapa soluautomaateille tyypillisessä ruudukossa tai vapaasti määritellyssä kolmiulotteisessa avaruudessa. Tämä vapaus ja joustavuus on yksi syy, miksi toimijapohjainen simulaatio on saavuttanut suurta suosiota monilla hyvin erilaisilla aloilla. Soluautomaatin määritelmä voisi edellä esitetyn perusteella olla siis vaikkapa seuraava: yksittäisten, vapaasti liikkuvien ja kommunikoivien, usein älykkäiden toimijoiden muodostama simulaatio, joka voidaan suorittaa käyttäjän määrittelemässä tilassa. Erityisen huomionarvoista toimijapohjaisessa simulaatiossa on, kuten soluautomaateissakin, mahdollisuus luoda hyvin monimutkaisia järjestelmiä, joista voi nousta esiin hyvinkin monimutkaisia malleja. Juuri siitä syystä toimijapohjaisia simulaatioita voidaan hyödyntää hyvin monenlaisissa tutkimuksissa, aina sosiaalisen kanssakäymi-

sen tutkimuksesta sotilasjoukon päätöksentekoon vaikuttavien tekijöiden etsimiseen.

Toimijapohjaisessa simulaatiossa yksittäisen toimijan älykkyydellä tarkoitetaan joitain ennalta määritettyjä sääntöjä tai algoritmeja, joiden perusteella yksittäiset toimijat toimivat. Toimijat voivat sisältää käyttäytymismalleja, ennalta määriteltyjä reaktioita tai käytännössä mitä tahansa muita esimerkiksi ihmisille tyypillisiä ominaisuuksia. Nämä ominaisuudet voidaan rakentaa monella eri tavalla tilanteesta riippuen, esimerkiksi vain yksittäisellä algoritmilla tai jopa kokonaisella neuroverkolla. Yksittäiset toimijat voivat tarvittaessa kommunikoida keskenään ja toimia saamiensa viestien perusteella. Kommunikaatio voidaan toteuttaa joko yksinkertaisella viestiperiaatteella, jossa toimija lähettää toiselle viestin, tai jollain hyvinkin kehittyneellä algoritmilla. Usein toimijoiden halutaan myös oppivan simulaation edetessä, jotta voitaisiin paremmin tutkia esiin nousevia käyttäytymismalleja, joita on havaittavissa esimerkiksi ihmispopulaatioissa. Tähän tarkoitukseen voidaan hyödyntää monia eri malleja, joita on kehitetty tutkimalla ihmisten oppimista.

Kuten aiemmin mainitsin, toimijapohjaista simulaatiota on käytetty hyvin laajalti useilla eri aloilla. Eräinä merkittävinä esimerkkeinä voidaan mainita esimerkiksi Yhdysvaltain merijalkaväen tutkimuslaitoksen kehittämä EINSTein [Ilachinski 2003], opetussovellus StarLogo [Kopfler and Begel 2003], sekä esimerkiksi tutkimus sosiaalisesta verkostoitumisesta ja sähköisestä kaupankäynnistä [Chan et al. 2010], sekä eräs tutkimus, jossa pyrittiin tutkimaan nykyisen evoluutiohistorian paikkaansapitävyyttä toimijapohjaisella simulaatiomallilla [Lafusa 2007]. Kaikissa tutkimuksissa käytetään joko toimijapohjaista simulaatiota, tai hieman sovellettua versiota siitä. EINSTeinin kehityksen taustalla oli pyrkimys ymmärtää ja simuloida aiempaa paremmin mahdollisia taistelutilanteita. Aiemmat simulaatiomallit perustuivat ensimmäisen maailmansodan ajalta oleviin malleihin vastapuolten voimasuhteista, jotka eivät sovellu moderniin, liikkuvaan ja hajallaan olevaan sodankäyntiin. Kehityksen tuloksena oli tietokoneohjelma, jonka simulaatiomallina toimii toimijapohjainen simulaatio yhdistettynä soluautomaattiin. Jokainen toimija on käytännössä yksittäinen sotilas, jolla on sotilasarvostaan riippuen tietynlainen joukko toimintatapoja, sekä jokaiselle joukon toimijalle yhteinen tavoite. Ohjelman avulla on mahdollista havaita eri joukkojen keskinäisestä toiminnasta esiin nousevia kollektiivisia toimintamalleja esimerkiksi hyökkäystilanteessa. Ohjelman avulla on mahdollista simuloida mielivaltaisia taistelutilanteita ja edelleen ymmärtää paremmin niin yksittäisten sotilaiden, kuin myös kokonaisten joukkuiden kollektiivista käyttäytymistä.

StarLogo syntyi tarpeesta ja halusta kehittää yksi ohjelma, jolla voidaan helposti tutkia mahdollisimman monenlaisia eri malleja ja teorioita. Aiemmat ohjelmat perustuivat lähinnä yksittäisen mallin simulointiin ja visualisointiin, eikä niitä voitu käyttää laaja-alaisesti esimerkiksi opetuksessa. Ohjelma rakentuu EINSTeinin tapaan toimijapohjaisen simulaation ja soluautomaattien yhdistelmästä, mutta sen lisäksi ohjelmaa varten on kehitetty oma virtuaalikone. Sen tarkoituksena on tehostaa simulaation toimintaa, sekä mahdollistaa soluautomaatin ja toimijoiden asynkroninen päivitys. Ohjelmassa on mahdollista ohjelmoida itse omia malleja käyttäen yksinkertaista Logo-ohjelmointikieltä. Näin ollen sitä voidaan käyttää jo hyvin alhaisilla opetustasoilla niin ohjelmoinnin, kuin monenlaisten eri mallienkin opetukseen. StarLogon eräänä tavoitteena olikin juuri yhdistää eri aineiden opetusta ja tehdä opetuksesta kysymyslähtöisempää (vrt. nykyinen, usein teorialähtöinen opetus); etenkin lapset oppivat paremmin kyselemällä ja kiinnostuvat helpommin päästessään itse tekemään ja vaikuttamaan.

Toimijapohjaista simulaatiota on käytetty paljon myös erilaisessa tutkimuksessa. Sosiaalista verkostoitumista ja sähköistä kaupankäyntiä on tutkittu hyödyntäen kyseistä simulaatiomallia [Chan et al. 2010]. Sosiaalista verkostoitumista varten kerättiin dataa opiskelijoiden käyttäytymisestä sosiaalisissa verkoissa. Tästä datasta yleistettiin joukko sääntöjä, joiden perusteella toimijat käyttäytyivät. Vaikka dataa kerättiin vain 30 opiskelijalta, tulokset sata kertaa isomman, simuloidun joukon käyttäytymisestä vastasivat todellisia tuloksia. Samankaltaista tapaa käytettiin myös sähköisen kaupankäynnin tutkimisessa sekä pitkäaikaisen evoluution mallintamisessa [Lafusa 2007]. Monet käytetyistä algoritmeista olivat jo aiemmin kehitetty, joten niitä vain sovellettiin tilanteen vaatimalla tavalla.

Vaikka toimijapohjaista simulaatiota on alettu käyttämään paljon hyvinkin erilaisilla aloilla, on se silti vielä tutkimusalueena melko hajallaan. Sitä käytetään hyvin erilaisiin tarkoituksiin eri aloilla, usein vieläpä hieman sovellettuna. Kunnollisia perussääntöjä tai malleja ei vielä ole kehitetty, eikä toimijapohjaisilla malleilla voida saavuttaa tieteellisesti yhdenmukaisia ja päteviä tuloksia. Toisaalta käytön yleistyessä on myös paljon todennäköisempää, että joku alkaa tutkia toimijapohjaisen simulaation teoriaa tarkemmin ja kehittää yleisesti hyväksytyt perusteoriat -ja säännöt.

5. Ongelmia

Tekoelämä on hyvin laaja käsite ja se kattaa monenlaisia teorioita ja tekniikoita, sekä myös kysymyksiä ja ongelmia. Sitä käytetään monin eri tavoin useilla eri

aloilla, vaikkakin se usein vain mainitaan yleisenä menetelmänä tai käsitteenä. Tekoelämällä voidaan ratkaista monia sellaisia ongelmia, joita ei voida tutkia tai testata joko eettisistä, ajallisista tai muista syistä. Kuitenkin tekoelämä pitää sisällään monia ongelmia, aina sen määritelmästä ja yhteydestä luonnolliseen elämään, kuin moniin hajallaan oleviin teorioihin ja menetelmiin.

Eräs vanhimmista ongelmista tulee jo von Neumannin teorioista. Hänen teoriansa monimutkaisesta itseään monistavasta koneesta synnytti monia keskusteluita teorioiden tärkeydestä ja merkittävyydestä. Jotkut olivat sitä mieltä, että von Neumannin teoriat olivat liian triviaaleja ja hänen ehdotuksensa itseään monistavasta koneesta liian monimutkainen. Kuitenkin ongelmallisinta niin von Neumannin kuin muidenkin ehdotuksissa oli monimutkaisuuden määritelmä. Von Neumann ei itse antanut tarkkaa, formaalia määritelmää monimutkaisuudelle. Tämä oli von Neumannin puolesta tahallista, sillä hänen monimutkaisen itseään monistavan koneensa määritelmä perustui osittain tähän hyvin löysään määritelmään. Hänen jälkeensä jotkut ovat yrittäneet määritellä monimutkaisuutta joko laskennallisten ominaisuuksien tai itse luodun kopion ominaisuuksien perusteella. Lisäksi on esitetty erilaisia mielipiteitä siitä, mikä on monimutkaisuuden vähimmäisvaatimus itseään monistavissa koneissa. Arthur Burks ehdotti vuonna 1970 vähimmäisvaatimukseksi kykyä universaaliin laskentaan, kun taas Langton esitti vuonna 1984, että itseään monistavan koneen tulisi koostua geeneistä ja ulkoasusta, jotka sen jälkeläisten tulisi periä ja muuttaa evoluution tapaan [McMullin 2000]. Kuitenkaan monimutkaisuutta tai itseään monistavan koneen vähimmäismonimutkisuutta ei ole vieläkään yksikäsitteisesti määritelty. Langtonin ehdotus on kuitenkin hyvin yleisesti käytetty, sillä se kuvastaa hyvin biologisten eliöiden rakennetta ja näin ollen myös luonnollista elämää [Sims 1994].

Toinen, ehkä tietojenkäsittelyn kannalta kiinnostavampi ongelma, on laskentateho. Vasta viime vuosien aikana tietokoneiden laskentateho on kasvanut sille tasolle, että jopa melko monimutkaisia simulaatioita voidaan suorittaa tavallisella PC:llä. Moniytimiset suorittimet ja monisäikeinen prosessointi mahdollistavat usean toimijan tai solun yhtäaikaisen päivittämisen, joka tuottaa yleisesti ottaen tarkempaa ja vähemmän virhealtista simulaatiota. Laskentatehon kasvusta huolimatta kaikista monimutkaisimmat ja paljon monimutkaisia toimijoita sisältävät simulaatiot vaativat usein hyvin suunniteltua ohjelmistoarkkitehtuuria. Virtuaalikoneet ovat hyvin yleinen ratkaisu, kun pyritään saavuttamaan simulaatio, jossa toimijoiden päivittäminen ei ole riippuvainen suorittimien jaksottaisesta rakenteesta. Ongelma on selvimmin esillä yksiytimisellä suorittimella; yksittäiset toimijat pitää päivittää järjestyksessä, sillä samanaikaisuutta ei juurikaan ole suorittimen puolesta saatavilla. Toinen ongelma lasken-

tatehon lisäksi onkin nykyisten Harvard-tyyppisten suoritinten arkkitehtuuri. Niissä suoritettava ohjelma ladataan muistiin kerran ja suoritetaan. Vaihtoehtoinen von Neumann-arkkitehtuuri sopisi paremmin juurikin soluautomaattien laskentaan, sillä ne tukevat suoritettavan ohjelmakoodin dynaamista muokkaamista samaan tapaan, kuin solukon yksittäisten solujen tilaa muokataan. Harvard-arkkitehtuuri on kuitenkin yleisempi, sillä se on tehokkaampi. Näin ollen kehittäjät joutuvat turvautumaan joko virtuaalikoneisiin tai laskennan jakamiseen suorittimen ja näytönohjaimen välille. Usein solujen tai toimijoiden päivitysjärjestyksen määrittämiseen käytetään myös algoritmia, joka määrittää päivitysjärjestyksen. [Langton 1986]

Näiden ongelmien lisäksi on myös filosofiset ja eettiset ongelmat. Eräs isoimmista filosofisista ongelmista on kysymys siitä, miten elämää voi esiintyä näennäisesti ei-elossa olevassa ympäristössä. Mikäli tietokone ei ole elossa oleva laite, miten sen sisällä voisi olla mitään elämään viittaavaa? Lisäksi on vielä kysymys siitä, mikä on mielen ja älykkyyden suhde elämään, ja voidaanko tekoelämän avulla luoda älykkäitä koneita. Mikäli oletetaan, että edellä mainitut ongelmat saataisiin yksimielisesti selvitettyä, ja tekoelämän päätettäisiin olevan niin sanottua aitoa elämää, esiintyy vielä eettiset kysymykset. Voidaanko tekoelämää hyödyntää tavoilla, jotka luonnollisessa elämässä ovat julmia tai epäeettisiä, mikäli tekoelämä on oikeaa elämää? Nämä ongelmat, sekä tekoelämän muutenkin melko epätieteellinen luonne ovat isoimpana esteenä sen käytölle tieteellisen tutkimuksen välineenä. Tekoelämätekniikoilla simuloidut tulokset eivät välttämättä ole kaikkien mielestä tarpeeksi tieteellisin menetelmin saavutettuja. [Bedau et al. 2000]

6. Hyötyjä

Tekoelämällä on ongelmistaan huolimatta paljon potentiaalia. Vaikka sen nimi viittaakin elämään ja eläviin olioihin, voidaan sillä mallintaa ja tutkia melkeinpä mitä tahansa, jopa elottomia asioita. Erityisesti toimijapohjaisen simulaation avulla voidaan helposti luoda monimutkaisia simulaatioita, joissa parhaimmillaan yhdistyy niin elottomat kuin elollisetkin oliot.

Opetuksessa tekoelämää voidaan käyttää jo melko varhaisesta vaiheesta aina korkeakoulutasolle. Lähes mikä tahansa malli ja teoria niin matematiikan, fysiikan, kemian, biologian kuin maantieteenkin alalta on mallinnettavissa. Varhaisopetuksessa tekoelämää voidaan käyttää, kunhan mallien luonti on toteutettu helpolla tavalla, eikä vaadi oppilailta paljon ennakkotietoa esimerkiksi ohjelmoinnista. Näin mahdollistetaan opetus opiskelijalähtöisesti; oppilaat saavat itse yrittää luoda malleja ja ideoida vapaasti. Tämän seurauksena oppimi-

sesta tulee hauskempaa ja innostavampaa, jolla taas on positiivinen vaikutus oppimiseen. [Kopfler and Begel 2003]

Tutkimuksessa tekoelämän avulla voidaan tutkia teorioita, joita ei ajallisesti tai eettisesti olisi mitenkään mahdollista tutkia. Esimerkiksi luonnonvalintaa ja sotien kehittymistä ei pystytä järkevästi tutkimaan ilman tekoelämän tarjoamia menetelmiä [Lafusa 2007]. Myös algoritmien tehokkuutta voidaan testata asettamalla ne luonnonvalintatilanteeseen muiden algoritmien kanssa, jossa niiden on selvittävä tai ”kuoltava”. Näin ollen voidaan helposti testata uusia algoritmeja ja arvioida niiden tehokkuutta olemassa olevia algoritmeja vastaan. [Gu et al. 2009].

7. Yhteenveto

Tutkielmassa käsittelin tekoelämää hyvin yleisellä tasolla ja vain muutaman tekniikan näkökulmasta. Käsittelin suppeasti tekoelämän historiaa ja määritelmiä, sekä esittelin muutaman keskeisessä roolissa olevan tekoelämätekniikan. Lopuksi kävin hieman läpi tekoelämän hyötyjä ja ongelmia.

Tutkielman aihe on laaja, mutta siitä ei siltikään ole kovin paljoa yleisestä näkökulmasta kirjoitettua kirjallisuutta. Pyrinkin tällä tutkielmalla tarjoamaan yleiskatsauksen tekoelämään. Toivon, että tulevaisuudessa aiheesta tehdään enemmän aloittelijoille soveltuvaa yleisen tason kirjallisuutta.

Viiteluettelo

Manuel Alfonseca. 1999. Programming Cellular Automata in APL2. *ACM SIGAPL APL Quote Quad* 30, 1, 27-30.

Mark A. Bedau, John S. McCaskill, Norman H. Packard, Steen Rasmussen, Chris Adami, David G. Green, Takashi Ikegami, Kunihiko Kaneko, Thomas S. Ray. 2000. Open problems in artificial life. *Artificial life.* 6. 4. 363-376.

Wai Kin Victor Chan, Young-Jun Son, Charles M. Macal. 2010. Agent-Based Simulation Tutorial – Simulation Of Emergent Behavior And Differences Between Agent-Based Simulation And Discrete-Event Simulation. In: *Proc. of The Winter Simulation Conference (WSC '10)*. 135-150.

Drue Coles. 2009. Artificial life as a path from computing to philosophy. *Journal of Computing Sciences in Colleges* 24, 6, 95-102.

Anthony H. Dekker. 1994. The game of life: a CLEAN programming tutorial and case study. *ACM SIGPLAN Notices* 29, 9, 91-114.

Yun-li Gu, Xin Xu, Jie Du and Hyan-yan Qian. 2009. Optimization algorithm based on artificial life algorithm and particle swarm optimization. In: *Proc.*

of Second International Conference on Information and Computing Science, 2009 (ICIC '09). 173-176.

Andrew Ilachinski. 2003. Exploring self-organized emergence in an agent-based synthetic warfare lab. In: Andrew Adamatzky. 2003. *Artificial Life*. Emerald Group Publishing. 38-76.

Eric Kopfler and Andrew Begel. 2003. StarLogo under the hood and in the classroom. In: Andrew Adamatzky. 2003. *Artificial life*. Emerald Group Publishing. 15-37.

Antonio Lafusa. 2007. Studying Long-term Evolution with Artificial Life. In: *Proc. of The 2007 IEEE Symposium on Artificial Life (CI-Alife 2007)*. 15-22.

C. G. Langton. 1986. Studying artificial life with cellular automata. *Physica D: Nonlinear Phenomena*, 22, 1, 120-149.

Barry McMullin. 2000. John von Neumann and the evolutionary growth of complexity: Looking backward, looking forward. *Artificial life*, 6, 4, 347-361.

Karl Sims. 1994. Evolving Virtual Creatures. In: *Proc. of the 21st Annual Conference on Computer Graphics and Interactive Tehcniques (SIGGRAPH '94)*.

Tu Xuyan. 2005. Life, artificial life and generalized artificial life. In: *Proc. of International Conference on Neural Networks and Brain, 2005 (ICNN&B '05)*. 1425-1428.

Tukivektorikone ilmalaserkeilausaineiston luokittelussa

Pasi Talvitie

Tiivistelmä

Laserkeilaus on mittausmenetelmä, jolla kolmiulotteisia kohteita voidaan mitata ja jonka pohjalta kohteista on mahdollista luoda kolmiulotteisia malleja. Ilma-aluksesta tehty ilmalaserkeilaus on muodostunut maanmittauksen perusmenetelmäksi, joka tuottaa suuria datamääriä. Aineiston käsittelyn oleellinen työvaihe on luokittelu, jossa aineiston sisältämä pistetieto jaetaan kuuluvaksi esimerkiksi maanpintaan tai kasvillisuuteen. Luokittelussa on hyödynnetty tilastollisia ja koneoppismenetelmiä.

Koneoppimismenetelmistä tukivektorikonetta on sovellettu laserkeilausaineistojen luokittelussa erityisesti luonnonvarasektorilla, jossa sitä on hyödynnetty metsävarojen inventoinnissa. Menetelmää on käytetty myös rakennetun ympäristön kohteiden luokittelussa ja luonnonmuodostelmiin liittyvässä riskikartoituksessa. Tukivektorikoneella on saavutettu hyviä tuloksia, mutta sen käyttö ovat jäänyt edelleen varsin vähäiseksi.

Tässä tutkielmassa luon katsauksen ilmalaserkeilauksen perusteisiin sekä tukivektorikoneen sovelluksiin ilmalaserkeilausaineistojen luokittelussa.

Avainsanat: Laserkeilaus, lidar, tukivektorikone, koneoppiminen

1 Johdanto

Laserkeilaus on kaukokartoitusmenetelmä, joka on viimeisen kymmenen vuoden aikana mullistanut maanmittausalan. Laserkeilaus on mahdollistanut laajojen alueiden yksityiskohtaisen mittaamisen tarkasti, nopeasti ja edullisesti. Sen sovelluksista erityisesti ilmalaserkeilaus (Airborne Laser Scanning, ALS) on otettu laajalti käyttöön sekä maastomallien että metsäarvioiden tekemisessä. Maalaserkeilaus puolestaan on muuttanut tapoja mallintaa esimerkiksi kaupunkiympäristöä, louhosalueita ja arkeologisia kohteita.

Laserkeilauksen varjopuolena on sen tuottama valtava tietomäärä. Kun keilainten pulssitaajuudet liikkuvat kymmenissä kilohertzeissä, kertyy havaittuja mittapisteitä tunnissa jopa miljardeja. Tämän tietomäärän tallentaminen ja käsittely on aikaa vievä prosessi, ja vaikka suuri osa työstä on automatisoitu, jää operaattorille vielä tehtäväksi miljoonien mittapisteiden läpikäynti käsin.

Laserkeilausaineiston käsittelyä voidaan nopeuttaa automatisoimalla osa sen työvaiheista. Tärkeimpiin automatisoitaviin työvaiheisiin kuuluu pisteiden luokittelu. Sovelluksesta riippuen aineisto luokitellaan yleensä kolmesta

seitsemään luokkaan [31]. Näihin sisältyvät maanpinta, kasvillisuus ja rakennukset. Kasvillisuus voidaan edelleen jakaa matalaan ja korkeaan kasvillisuuteen, ja rakennetussa ympäristössä voidaan huomioida myös tiet, ajoneuvot, aidat, rautatiet ja sähkölinjat. Erityissovelluksissa luokittelu voidaan viedä vielä pidemmälle, esimerkiksi pyrkimällä erottamaan eri puulajit toisistaan [16].

Luokittelussa sovelletaan yleisesti päätöspuita, neuroverkkoja ja suurimman todennäköisyyden menetelmää [19, 22]. Näille menetelmille on yhteistä, että ne vaativat opetusvaiheessa melko suuren opetusjoukon. Näistä poikkeava luokittelussa hyödynnetty menetelmä on tukivektorikone (Support Vector Machine, SVM), joka pystyy saavuttamaan perinteisten menetelmien suorituskyvyn huomattavasti pienemmällä opetusdatamäärällä. Tämä nopeuttaa luokitteluprosessia ja vähentää myös hitaiden ja kalliiden maastotarkistusten määrää.

Tutkielman rakenne on seuraava: Luvussa 2 esitellään laserkeilaus ja luvussa 3 laserkeilausaineistojen piirreavaruudet. Luku 4 esittelee tukivektorikoneen sekä sen sovelluksia ilmalaserkeilausaineistojen käsittelyssä. Luvussa 5 on yhteenveto ja pohdintaa.

2 Laserkeilaus

Laser (light aplification by stimulated emission of radiotion) on keinotekoisesti aikaansaatua hyvin jäsentynyttä ja järjestäytynyttä valoa. Laservalo on koherenttia eli sen aallot ovat kaikki samassa vaiheessa. Laserilla saadaan aikaan tarkasti suunnattavissa oleva valonsäde, ja säteen voimakkuudesta johtuen sen osuminen kohteeseen voidaan havaita hyvin kaukaa.

Laserkeilaus on menetelmä, joka perustuu kohteeseen lähetetyn ja sieltä takaisin sironneen lasersäteen kulkuajan mittaukseen. Kulkuajan kaava on

$$t_L = 2\frac{R}{c}, \tag{2.1}$$

jossa t_L on lasersäteen kulkema aika, R etäisyys keilaimen ja kohteen välillä sekä c valon nopeus [30]. Kohteen etäisyys saadaan vastaavasti kaavalla

$$R = \frac{1}{2}c \cdot t_L. \tag{2.2}$$

Säteen kulkuajan lisäksi on tiedettävä lähettimen asento ja säteen lähtökulma ennen kuin voidaan laskea kohteen etäisyys lähettimestä, jonka perusteella voidaan laskea mitatun kohteen sijainti kolmiulotteisessa avaruudessa suhteessa lähettimen sijaintiin [30]. Laserkeilaukseen voidaan käyttää joko pulssilaseria tai jatkuva-aaltoista laseria. Käytännössä suurin osa tuotantokäytössä olevista ilmalaserkeilaimista hyödyntää pulssilaseria, jossa yksittäisen laserpulssin ajallinen pituus on noin 10 ns [30]. Jatkuva-aaltoisia lasereita hyödynnetään kaupallisesti etupäässä maalaserkeilauksessa [25].

Laserkeilauksella mitatun pisteen sijaintitiedon lisäksi takaisinsiroava säteily sisältää intensiteettitiedon eli tiedon kohteesta siroavan säteilyn voimakkuudesta. Koska kaupalliset laserit hyödyntävät yleensä lähi-infrapunaalueen (1064 - 1550 nm aallonpituus) lasereita, voidaan intensiteettitiedon spektrin perusteella erottaa myös, minkä tyyppisestä kohteesta pulssi on sironnut [31].

Lasersäteen pinta-ala maanpinnalla voidaan laskea kaavalla

$$A_I = \frac{\pi}{4}(D + R\gamma)^2, \tag{2.3}$$

jossa A_I on lasersäteen maan pinnalla valaisema pyöreän alueen pinta-ala, D on laserin aukko eli apertuuri, R etäisyys ja γ kulmaero (angular divergence; tässä kaavassa γ:lla ei yksikköä) [30]. Kulmaero viittaa lasersäteen tulokulman erotukseen suorasta kulmasta; pystysuora lasersäde valaisee ympyrän muotoisen alueen, mutta kulman poiketessa pystysuorasta tuloksena on ellipsin muotoinen alue. Lasersäteen laajuus maanpinnalla voidaan esittää myös laserjalanjälkenä A_L, joka on yhtä kuin ympyrän muotoisen lasersäteen halkaisija eli $A_L = 2\sqrt{\frac{A_I}{\pi}}$. Osumakohdan sijainniksi lasketaan säteen peittoalueen keskipiste.

Laserkeilauksen tarkkuus pulssilaseria käytettäessä voidaan laskea kaavalla

$$\sigma_{R_{pulssi}} \sim \frac{c}{2} t_{nousu} \frac{\sqrt{B_{pulssi}}}{P_{R_{huippu}}}, \tag{2.4}$$

jossa R on etäisyys, c valon nopeus, t_{nousu} pulssin nousuaika, B_{pulssi} syötteen kohinan taajuus sekä $P_{R_{huippu}}$ laserpulssin optisen tehon huippuarvo [30].

Laserkeilauksen suosio maanmittauksessa perustuu sen tarkkuuteen, edullisuuteen ja kykyyn läpäistä esimerkiksi puiden latvustoa ja muita kasvillisuuden aiheuttamia näköesteitä. Mittauksen suuren taajuuden ja erityisesti ilmalaserkeilauksessa laserpulssin suurehkon pinta-alan vuoksi aina osa pisteistä lävistää kasvillisuuden ja siroaa takaisin vasta halutusta kohteesta, mikä mahdollistaa muun muassa maanpinnan tarkan mittaamisen puuston läpi ilma-aluksesta käsin.

2.1 Laserkeilausmenetelmien jaottelu

Suuren mittakaavan kohteisiin kohdistuva laserkeilaus jaetaan karkeasti kolmeen ryhmään: maalaserkeilaukseen, mobiiliin laserkeilaukseen ja ilmalaserkeilaukseen [15]. Jaottelu perustuu keilaimen alustaan. Eri alustatyypit vaativat erilaisten asioiden huomioonottamista, ja liikkuvissa alustoissa on ratkaistava myös keilaimen sijainti- ja asentotietoon liittyvät ongelmat [30].

Maalaserkeilauksessa (Terrestrial Laser Scanning, TLS) keilain on asetettu kiinteästi mittauspisteeseen, esimerkiksi kolmijalalle, josta käsin keilain

skannaa mitattavan kohteen [25]. Useammasta mittauspisteestä mittaamalla pystytään mittaamaan kohde siten, että mahdollisista esteistä johtuvia katveita ei tule. Maalaserkeilausta käytetään yleisesti esimerkiksi kaupunkiympäristössä rakennusten mittaamiseen. Maalaserkeilauksessa päästään tyypillisesti alle senttimetriluokan tarkkuuksiin, sillä mitattavat kohteet ovat useimmiten enimmillään joidenkin kymmenien metrien päässä keilaimesta.

Mobiili laserkeilaus on maalaserkeilauksen sovellus, jossa keilain on asennettu liikkuvalle alustalle, yleensä autoon tai muuhun ajoneuvoon [25]. Keilaimen lisäksi mittauksessa tarvitaan paikannusjärjestelmää, jonka tuottaman paikkatiedon perusteella ajoneuvon sijainti voidaan laskea. Mobiilikeilaimia käytetään esimerkiksi tieverkoston ja siihen liittyvien kohteiden mittaamiseen. Mobiilikeilauksessa päästään parhaimmillaan senttimetriluokan tarkkuuksiin, sillä maalaserkeilauksen tapaan mitattavat kohteet ovat useimmiten enimmillään joidenkin kymmenien metrien päässä keilaimesta. Mobiilikeilauksen erityisryhmä ovat itse tienpinnan mittaukseen tarkoitetut sovellukset [10].

Ilmalaserkeilaus (Airborne Laser Scanning, ALS) on ilma-aluksesta tehtävää laserkeilausta [30], jossa kohde keilataan ylhäältä päin. Myös ilmalaserkeilauksessa on huomioitavan keilaimen sijaintiin ja asentoon liittyvät parametrit. Keilausetäisyydet ovat kuitenkin huomattavasti suurempia kuin tavanomaisessa mobiilissa keilauksessa, mistä johtuen virhelähteitä on enemmän. Ilmalaserkeilaimen alustana on tyypillisesti joko lentokone tai helikopteri (kuva 1).

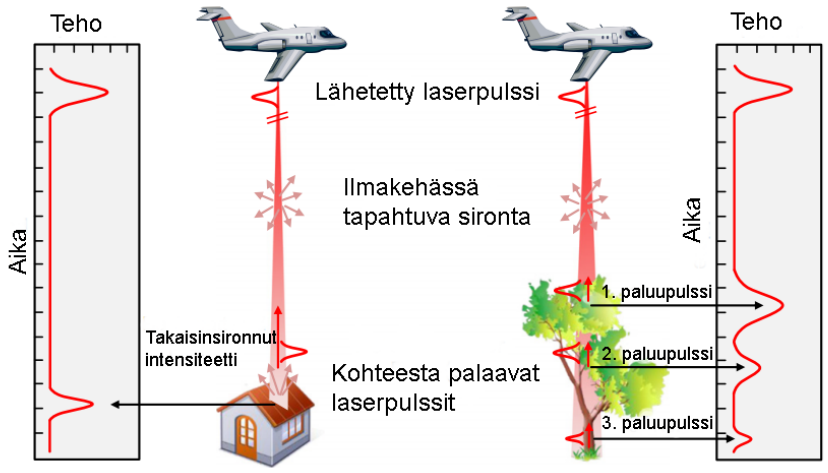
Ilmalaserkeilauksella pystytään kattamaan nopeasti laajojakin alueita, ja sitä käytetään yleensä maanmittauksen kartta-aineiston tuotannon pohjana. Se on myös immuuni valaistusolosuhteissa tapahtuville muutoksille, esimerkkinä valon ja varjojen voimakas vaihtelu, sekä perspektiiville, joka ilmavalokuvauksessa aiheuttaa katveita ja vääristymiä [31]. Ilmalaserkeilauksen tarkkuus riippuu keilaimen mittaustaajuudesta ja mittauskorkeudesta, mutta on tyypillisesti desimetriluokkaa. Keilauskorkeus voi sovelluksesta riippuen sadoista metreistä jopa yli kilometriin.

Ilmalaserkeilauksen uusimpana sovelluksena ovat miehittämättömien ilmaalusten (Unmanned Aerial Vehicle, UAV) käyttö mittausalustana [17]. Edullisia ja tehokkaita sähkökäyttöisiä koptereita hyödyntäen voidaan keilaus kohdistaa helposti ja nopeasti haluttuun kohteeseen. Alhaisten mobilisaatiokustannusten lisäksi mittauksen tarkkuus on parempi kuin suurista korkeuksista lentokoneesta tehdyissä mittauksissa.

2.2 Ilmalaserkeilainten virhelähteet

Mobiilien keilainten haasteet liittyvät sijainti- ja asentotietoon sekä keilaimen ja paikannusyksikön väliseen kalibraatioon. Sijantitieto saadaan GPS-yksiköstä (Global Positioning System, GPS), jonka tuottaman sijaintitiedon taajuus on kuitenkin huomattavasti alhaisempi kuin laserkeilaimen. Tästä

Kuva 1: Ilmalaserkeilauksen toimintaperiaate. Vasemmalla tilanne, jossa kaiku tulee yksittäisestä kohteesta, kuten rakennuksesta. Oikealla tilanne, jossa useita kaikuja puustoisesta kohteesta. Pystydiagrammit kuvaavat kaikujen sijaintia ja intensiteettiä. [31]



johtuen sijainti määritetään GPS-mittausten välisen ajan osalta laskennallisesti inertiamittausyksikön (Inertial Measurement Unit, IMU) tuottaman jatkuvuustiedon perusteella hyödyntäen Kalman-suodatusta [10, 26].

Ilmalaserkeilauksessa sijainnin lisäksi on otettava huomioon keilaimen asento, eli lentokoneen kallistus ja mahdollinen sorto eli koneen pituusakselin poikkeama lentosuunnasta. Koska keilain mittaa ilma-aluksesta alaspäin, sijoitetaan se aluksen alaosaan. GPS ja IMU sijaitsevat puolestaan aluksen yläosassa, jotta yhteys satelliitteihin olisi mahdollisimman esteetön. Keilaimen ja GPS:n sekä IMU:n välisen sijainnin kalibrointiin vaikuttavat puolestaan esim. materiaalien käyttäytyminen eri lämpötiloissa. Lentokoneesta laserilla mitattaessa on otettava huomioon myös ilman suhteellinen kosteus, ilmanpaine ja lämpötila, jotka vaikuttavat valon nopeuteen ja sitä kautta sijaintitiedon tarkkuuteen.

2.3 Ilmalaserkeilausaineiston käsittely

Ilmalaserkeilausaineiston käsittelyssä on useita työvaihteita. Työnkulun käytännöt on seuraavassa referoitu Soinisen ja Korpelan [28] mukaisesti.

Mittauksen jälkeen ensimmäinen vaihe on aineiston pilkkominen hallittaviin osiin. Tyypillisessä ilmalaserkeilausprojektissa mitattujen pisteiden määrä voi vaihdella 5 ja 50 miljardin pisteen välillä, joka täytyy jakaa pienempiin osiin tietokoneella tehtävää käsittelyä varten. Yleensä noin 5 - 20 miljoonan pisteen suuruisiin osiin jakaminen mahdollistaa aineiston tehokkaan jatkokä-

sittelyn. Datan organisoinnin ja koordinaatistoon rekisteröinnin jälkeen data kalibroidaan, mikä sisältää muun muassa mittauslaitteistosta johtuvien virheiden ja korkeuskorjauksen sekä eri lentojonojen yhteensovittamisen pintojen sovituksen avulla.

Alkuvalmisteluiden jälkeen pisteet luokitellaan. Tästä 90% tapahtuu automaattisesti, mutta miljoonien datapisteiden ollessa kyseessä jäljelle jäävä 10% muodostaa vielä huomattavan pistemäärän käsin luokiteltavaksi. Sovelluksesta riippuen aineisto luokitellaan kolmesta (maanpinta, kasvillisuus, muut maanpinnan yläpuoliset kohteet) jopa yli kymmeneen luokkaan. Tyypillisiä luokiteltavia kohteita ovat edellä mainittujen lisäksi rakennukset, tiet ja muu pitkittäinen infrastruktuuri, ajoneuvot, erilaiset luonnonmuodostelmat, kasvillisuuden alalajit kuten ruoho, pensaat sekä puusto.

Maanpinnan luokittelussa käytetään usein algoritmeja, jotka perustuvat pistepilven alimpien pisteiden etsintään [27]. Esimerkiksi Axelssonin [2] algoritmissa pistepilven alimpien pisteiden perusteella muodostetaan löyhä kolmioverkkopintamalli, jota iteratiivisesti tarkennetaan lisäämällä malliin kunkin kolmion sisältä piste, joka on tarpeeksi lähellä sekä pintaa että kolmion muodostavia pisteitä. Pisteiden lisäyksen jälkeen kolmioverkkopintamalli lasketaan uudelleen ja prosessia toistetaan, kunnes malliin ei enää voida lisätä pisteitä.

Maanpinnan yläpuolisten kohteiden luokittelussa menetelmien kirjo on laaja ja riippuu kulloinkin tehtävän luokittelun tavoitteesta. Luokittelussa hyödynnetään datan segmentointia ja usein myös klusterointia [26]. Kasvillisuuden luokittelussa tutkittavat parametrit poikkeavat esimerkiksi rakennetun ympäristön luokittelusta. Kasvillisuudelle on tyypillistä kaikujen suuri hajonta syvyyssuunnassa, kun taas rakennetussa ympäristössä kaiut keskittyvät usein tasopinnoille, kuten talojen katoille tai tien pinnoille. Yksittäisen menetelmän sijaan käytetäänkin usein yhdistelmämenetelmiä, ja lähtöaineistona laserkeilauksen xyz-koordinaattitiedon ohella intensiteettitietoa tai samalta alueelta rekisteröityjä ilmakuvia tai hyperspektrikuvia [31].

Erityissovelluksissa voidaan pyrkiä luokittelemaan tiettyjä kohteita tarkemmin. Esimerkiksi puuston osalta puulajitieto voidaan määrittää [16] ja puuston ominaisuuksien perusteella voidaan myös arvioida metsästä saatavaa puumäärää kuutiometreinä [23]. Suomessa yksi laserkeilauksen tärkeimmistä sovelluksista onkin metsävarantoarvioiden tekeminen, jossa laserkeilauksen avulla voidaan tuottaa tarkempaa tietoa kuin perinteisin, osin subjektiivisin menetelmin [16, 23].

Mikäli keilauksen yhteydessä on otettu myös ilmakuvat, ne rekisteröidään keilausaineiston kanssa samaan koordinaatistoon. Ilmakuvien avulla voidaan myös validoida keilauksen laatu ja tarvittaessa tehdä sijaintiin korjauksia. Monissa analyysimenetelmissä ilmakuvia voidaan hyödyntää myös aineiston luokittelussa, kuten esimerkiksi Yan ja muut [31] esittävät.

Aineiston käsittelyn viimeinen vaihe on aineiston tilaajalle lähtevien lopputuotteiden valmistaminen. Vähimmillään tämä sisältää raakadatan lisäk-

si kolmioverkkopintamallin valmistuksen (Triangulated Irregular Network, TIN). Pintamallilla tarkoitetaan yleensä maanpinnan korkeusmallia (Digital Elevation Model, DEM) tai maanpinnan, kasvillisuuden sekä muut pinnan yläpuoliset kohteet sisältävää maastomallia (Digital Surface Model, DSM). Joissain erikoistapauksissa lopputuotteet voivat keskittyä pelkkiin pinnan yläpuolisiin kohteisiin. Esimerkiksi normalisoitu pintamalli (Normalised Digital Surface Model, nDSM) on malli, josta maanpinnan vaihtelut on poistettu. Tiettyjen luokkien edustajista voidaan myös luoda erilliset pintamallit, kuten esimerkiksi puista tai rakennuksista.

3 Laserkeilausaineiston piirreavaruudet

3.1 Korkeustieto

Ilmalaserkeilauksen tuottamasta datasta tärkein on mitatun kohdepisteen korkeustieto eli z-koordinaatti [31]. Sen avulla voidaan erottaa sekä maanpinta että pinnan yläpuoliset kohteet. Korkeustiedon pohjalta voidaan luoda halutunlainen digitaalinen pintamalli.

Korkeustietoa voidaan hyödyntää myös muiden aineistojen tulkinnasta, esimerkkinä ilmakuvat ja hyperspektrikuvat. Tällöin erityisesti eri kasvillisuustyppien erottamistarkkuus paranee [11].

Korkeustiedon johdannainen on rinteen jyrkkyystieto (slope). Tätä voidaan hyödyntää esimerkiksi sähkölinjojen tunnistamisessa [31]. Lisäksi pisteiden jakauman vinouden (skewness) ja huipukkuuden (kurtosis) avulla voidaan esimerkiksi erottaa istutettu metsä luontaisesta metsästä [1].

3.2 Intensiteettitieto

Laserkeilauksen radiometrinen komponentti eli intensiteettitieto kertoo mitatun pinnan heijastusominaisuuksista ja sitä kautta pinnan rakenteesta ja pintamateriaalista [31]. Sitä voidaan hyödyntää luokittelussa. Imin ja muiden [13] mukaan intensiteettitietoa hyödyntämällä saavutetaan 10-20% parempi luokitteluteho.

Intensiteettitiedossa on aina jonkin verran kohinaa, mikä heikentää sen käyttökelpoisuutta tietyissä sovelluksissa [31]. Kohinan minimointi on toisaalta osa laitteiston kalibrointia, tulosten korjauksia ja normalisointia.

3.3 Monikaiut ja tekstuuri

Pulssilaserkeilainta käytettäessä tallennetaan mittauspistettä kohden yleensä ensimmäinen ja viimeinen kaiku. Ensimmäinen kaiku edustaa maanpinnan yläpuolisia kohteita ja viimeinen maanpintaa. Monikaikuja syntyy erityisesti

ilmalaserkeilauksessa, sillä mittauskorkeudesta riippuen laserpulssin halkaisija maanpinnan lähellä voi vaihdella noin kymmenestä senttimetristä useampiin metreihin [23]. Tällöin osa pulssista siroaa takaisin jo puiden latvuksista osan jatkaessa matkaansa oksiston läpi maahan asti [31].

Ensimmäisen ja viimeisen kaiun välisen etäisyyden vaihtelua ja kaikujen intensiteettieroja voidaan hyödyntää luokittelussa. Eri intensiteettiarvojen yhteisesiintymismatriisin (Gray-Level Co-Occurrence Matrix, GLCM) avulla voidaan arvioida mitatun pinnan tekstuuria, millä saattaa olla merkittävä rooli erityisesti tukivektorikoneen avulla tehtävässsä luokittelussa [12].

3.4 Useasta lähteestä peräisin olevan mittaustiedon yhdistäminen

Koska laserkeilaimen tuottama spektritieto on rajallinen, tulosta voidaan täydentää käyttämällä hyväksi muista sensoreista peräisin olevaa mittaustietoa. Esimerkiksi ilmakuvista voidaan erottaa kutakin mittauspistettä vastaava RGB-väriarvo. Tämä kuitenkin edellyttää, että ne on rekisteröity samaan koordinaattijärjestelmään ja niiden resoluutio on sama tai lähellä laserkeilauksen resoluutiota. [31]

Monisensoridatan hyödyntäminen soveltuu erityisesti laajojen alueiden maankäytön tarkasteluun. Ilmakuva- ja hyperspektriaineistoja voidaan hyödyntää esimerkiksi kasvillisuuden tai rakennetun ympäristön kohteiden luokittelussa [6, 31].

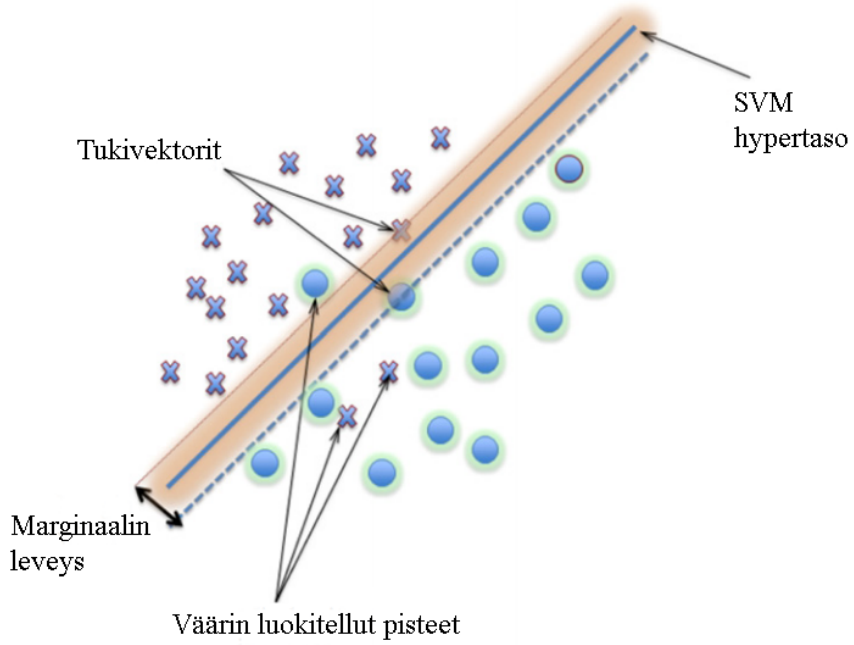
4 Tukivektorikone ja sen sovellukset laserkeilausaineiston luokittelussa

4.1 Tukivektorikone

Tukivektorikone on ohjattu ei-parametrinen tilastollinen oppimismenetelmä, jota käytettäessä ei tehdä oletuksia käsiteltävän datan tilastollisesta jakaumasta [22]. Menetelmä perustuu Vapnikin ja Chernonenkisin 1960-luvulla tekemään työhön, mutta laajempaa mielenkiintoa se on alkanut herättämään vasta 1990-luvulla [3]. Menetelmässä joukko tietoalkioita pyritään luokittelemaan kahteen ryhmään alkioiden ominaisuuksien perusteella. Yksinkertaisimmillaan lineaarinen binäärinen tukivektorikone pyrkii opetusaineiston avulla hakemaan funktion, jolla voidaan löytää ryhmät erottava pinta eli optimaalinen erottava hypertaso. Kyseessä on joukon sisältämät luokat erottava päätösrajapinta, joka minimoi väärin luokiteltujen alkioiden määrän ja maksimoi luokittelussa käytettyjen alkioiden ja hypertason välisen marginaalin [21, 24].

Tukivektorikone hyödyntää luokittelussa vain osaa luokiteltavan joukon näytepisteistä. Hyödynnettävät pisteet ovat ne, jotka sijaitsevat lähimpänä

Kuva 2: Lineaarisen tukivektorikoneen toimintaperiaate. Kahta luokkaa erottava hypertaso maksimoi tukivektoreiden välisen marginaalin ja minimoi väärin luokiteltujen pisteiden määrän. Mountrakis ja muut [22] Burgesin [3] pohjalta.



joukkojen välistä rajapintaa. Näitä pisteitä kutsutaan tukivektoreiksi [24]. Tämä ominaisuus keventää menetelmän tuottamaa laskennallista taakkaa, kun luokittelussa ei tarvita kaikkia näytepisteitä.

Tukivektorikoneen oppimisvaihe on iteratiivinen prosessi, jossa opetusjoukon avulla etsitään luokittelijaa, joka tuottaa joukon eri luokat erottavan optimaalisen päätöstason. Kun luokittelija on löydetty, sitä sovelletaan varsinaiseen aineistoon [32]. Luokittelija on niin sanottu ydinfunktio [4].

Tukivektorikone on yleistävä metodi, joka ei aina käytä kaikkia opetusjoukon alkoita. Päähuomio opetusjoukossa on alkiotyyppien rajapinnan läheisissä pisteissä, jotka muodostavat opetusjoukon alijoukon eli tukivektorin, jota käytetään hyväksi maksimaalisen erottavan hypertason etsinnässä (kuva 2) [22].

Tukivektorikoneen toiminta voidaan esittää seuraavasti [21]: Oletetaan, että opetusjoukko koostuu N vektorista d-ulottuvuuksisessa piirreavaruudessa $x_i \in R^d (i = 1, 2, ..., N)$. Jokaiseen vektoriin x_i liittyy tavoite $y_i \in \{-1, +1\}$. Oletetaan, että joukossa on kaksi luokkaa, jotka voidaan lineaarisesti erottaa. On siis olemassa luokkia erottava hypertaso, joka voidaan esittää hypertason normaalivektorilla $w \in R^d$, sekä kynnysarvo $b \in R$, joilla luokat voidaan erottaa ilman virheitä.

Se, kumpaan luokkaan joukon alkiot kuuluvat, voidaan määrittää funktion sgn[f(x)] avulla, jossa f(x) on hypertasoon liittyvä erotusfunktio, joka määritellään

$$f(x) = w \cdot x + b. \tag{4.1}$$

Luokkia erottavan hypertason löytämiseksi täytyy määrittää sellaiset w ja b, että

$$y_i(w \cdot x_i + b) > 0, \tag{4.2}$$

jossa $i = 1, 2, ..., N$.

Lineaarista tukivektorikonetta käytettäessä oletetaan, että luokiteltava joukko on erotettavissa lineaarisesti. Tällainen tilanne on kuitenkin todellisuudessa harvinainen, ja yleensä luokiteltavien joukkojen alkiot eivät ole erotettavissa lineaarisen pinnan avulla. Lineaarisen luokittelijan sijaan käytetäänkin usein muita menetelmiä, kuten joustavan marginaalin metodia [7], jossa erottamattomuusongelma ratkaistaan lisäämällä ylimääräisiä muuttujia tukivektorikoneen optimointiin. Toinen vaihtoehto on käyttää ns. kerneltemppua, jossa alkuperäisessä piirreavaruudessa olevat havainnot kuvataan korkeampiulotteisessa avaruudessa, jossa luokat voidaan erottaa lineaarisesti.

Tukivektorikone vaatii ydinfunktiolta, että se täyttää Mercerin teoreeman [4]. Ydinfunktion valinnalla on olennainen merkitys lopputuloksen kannalta. Mikäli sovellusalueen piirreparametrien käyttäytymisestä ei ole riittävää tietoa, voidaan käyttää esimerkiksi yleiskäyttöistä Gaussin ydinfunktiota [20].

Yleensä kaukokartoitusmenetelmien luokittelussa pyritään tunnistamaan useampia kuin kaksi luokkaa. Binääristä tukivektorikonetta voidaan tällöin muokata siten, että sitä voidaan käyttää moniluokkaisena luokittelijana, käyttäen esimerkiksi yksi-kaikkia-vastaan -periaatetta. Tällöin aineisto käydään läpi luokka kerrallaan, aina haettavaa luokkaa muita luokkia vastaan verraten [22]. Kyseessä on siis useiden binääritukivektorikoneiden yhdistelmä.

Tukivektorikoneella on monia hyviä ominaisuuksia. Dalponten ja Bruzzonen [8] mukaan tukivektorikoneen tärkeimmät ominaisuudet ovat 1) sen korkea luokittelutarkkuus sekä hyvä yleistävyys, 2) arkkitehtuurin suunnittelun ja opetusvaiheen suhteellinen helppous muihin koneoppimismenetelmiin verrattuna, 3) kustannusfunktion konveksisuus, mikä takaa aina optimituloksen löytymisen sekä 4) tehokkuus tilanteissa, joissa opetusjoukon koko on pieni suhteessa luokiteltavan aineiston kokonaismäärään.

Tukivektorikone pyrkii myös minimoimaan näkemättömän datan luokitteluvirheen ilman, että siihen liittyy oletuksia aineiston todennäköisyysjakaumasta. Useissa tilastollisissa menetelmissä sen sijaan oletetaan, että jakauma tunnetaan [22]. Fauvelin ja muiden mukaan [9] Gaussin jakaumaa noudattelevien funktioiden käyttäminen kaukokartoitusdatan luokittelussa ei ole

järkevää, sillä todellisuudessa mitattujen aineistojen jakaumat noudattavat harvoin normaalijakaumaa. Yleensä datassa on tihentymiä jakauman äärilaidoilla.

Tukivektorin huonoja puolia ovat menetelmän hitaus etenkin varsinaisessa luokitteluvaiheessa [3]. Tukivektorikone ei myöskään ota huomioon vierekkyyttä, mistä johtuen esimerkiksi järven keskellä oleva piste on mahdollista luokitella virheellisesti asfaltoiduksi maantieksi, vaikka ympärillä on pelkkää vettä [31].

4.2 Tukivektorikoneen sovellukset laserkeilausaineiston luokittelussa

Tukivektorikonetta on sovellettu laserkeilausaineistojen luokittelussa [16], mutta yleisemmin aihepiiriä on tarkasteltu kaukokartoitusaineistojen luokittelun näkökulmasta [22, 24]. Kaukokartoitusaineistoissa tukivektorikonetta on hyödynnetty muun muassa satelliittikuvien luokittelussa, mutta erityisen kiinnostuksen kohteena ovat olleet hyperspektrikuvien luokittelusovellukset, sillä hyperspektriaineistolle on ominaista aineiston ulottuvuuksien suuri määrä [22]. Kuitenkin myös laserkeilausaineiston luokittelussa voidaan hyödyntää tukivektorikoneen ominaisuuksia, kuten esimerkiksi Lin ja Hyyppä [16] esittävät. Vaikka laserkeilaus menetelmänä tuottaa vain rajallisen määrän piireavaruuksia, on näistä piirreavaruuksista mahdollista laskea lukuisa määrä erilaisia muuttujia, jotka nostavat luokiteltavien ulottuvuuksien määrää huomattavasti.

Yleisimmät tukivektorikoneiden sovellukset ilmalaserkeilausaineiston käsittelyssä liittyvät metsätalouteen tai metsäekologiaan. Puulajien tunnistaminen esitellään ensimmäisenä sovellusesimerkkinä. Tukivektorikoneita on myös sovellettu rakennetun ympäristön kohteiden luokittelussa, mikä käsitellään toisena esimerkkinä. Kolmas esimerkkisovellus on maanvyöryriskialueiden paikallistamiseen liittyvien tekijöiden tunnistaminen ilmalaserkeilausaineistosta.

Aineiston luokittelun ohella tukivektorikonetta voidaan myös soveltaa kohinan poistoon laserkeilausaineistosta [29]. Menetelmällä voidaan parantaa laserkeilauksen tarkkuutta erityisesti pitkillä matkoilla, joissa signaali/kohina -suhde putoaa voimakkaasti matkan kasvaessa.

4.3 Esimerkki 1: Puulajien tunnistaminen

Puulajien tunnistaminen on digitaalisessa metsätaloudessa oleellisen tärkeää. Ilmalaserkeilauksen yleistyttyä mahdollisuudet metsän ominaisuustiedon hankintaan ovat parantuneet, mutta toistaiseksi tiedon hyödyntäminen on ollut vajavaista. Useat menetelmät pyrkivät tunnistamaan, onko kyseessä havu- vai lehtipuu, mutta lajitasolle päästään harvoin. Lajitieto on kuitenkin

tärkeä tieto, ja ilman sitä laserkeilausaineiston pohjalta tehtyihin päätöksiin liittyy virhepäätelmien ja -arvioiden riski. [16]

Puulajien tunnistamista kaukokartoitusaineistosta tukivektorikoneen avulla ovat käsitelleet Melgani ja Bruzzone [21], jotka sovelsivat menetelmää hyperspektrikuvien pohjalta tapahtuvaan puulajien luokitteluun. Hyperspektrikuvissa luokittelun pohjana käytetään hyperdimensionaalista aineistoa, jossa luokittelun pohjana olevia ulottuvuuksia voi olla suuri määrä. Tällöin luokittelijana käytetyn menetelmän tehokkuus ja opetusaineiston pienuus ovat tärkeitä ominaisuuksia.

Lin ja Hyyppä [16] esittävät menetelmän puulajien automaattiseksi tunnistamiseksi tukivektorikoneen avulla. Käytetty tukivektorikone perustui LIBSVM[1]-pakettiin [5]. Menetelmässä hyödynnettiin neljää eri piirreparametrikategoriaa: laserpisteiden jakaumaa, laserkaiun intensiteettiä, latvuksen sisäistä rakennetta sekä puun ulkoista rakennetta. Kuhunkin kategoriaan kuului 9 - 13 eri piirreparametriä. Tavoitteena oli löytää optimaalinen piirreparametriyhdistelmä testattavan alueen puulajien tunnistamiseksi. Koealueena toimi Helsingin Seurasaaren luonnontilainen puistometsä, josta pyrittiin tunnistamaan neljä yleisintä puulajia: kuusi (*Picea Abies*), mänty (*Pinus Silvestris*), haapa (*Populus Tremula*) ja tammi (*Quercus Robur*). Metsästä valittiin 40 edellä mainittujen puulajien edustajaa luokittelukoetta varten.

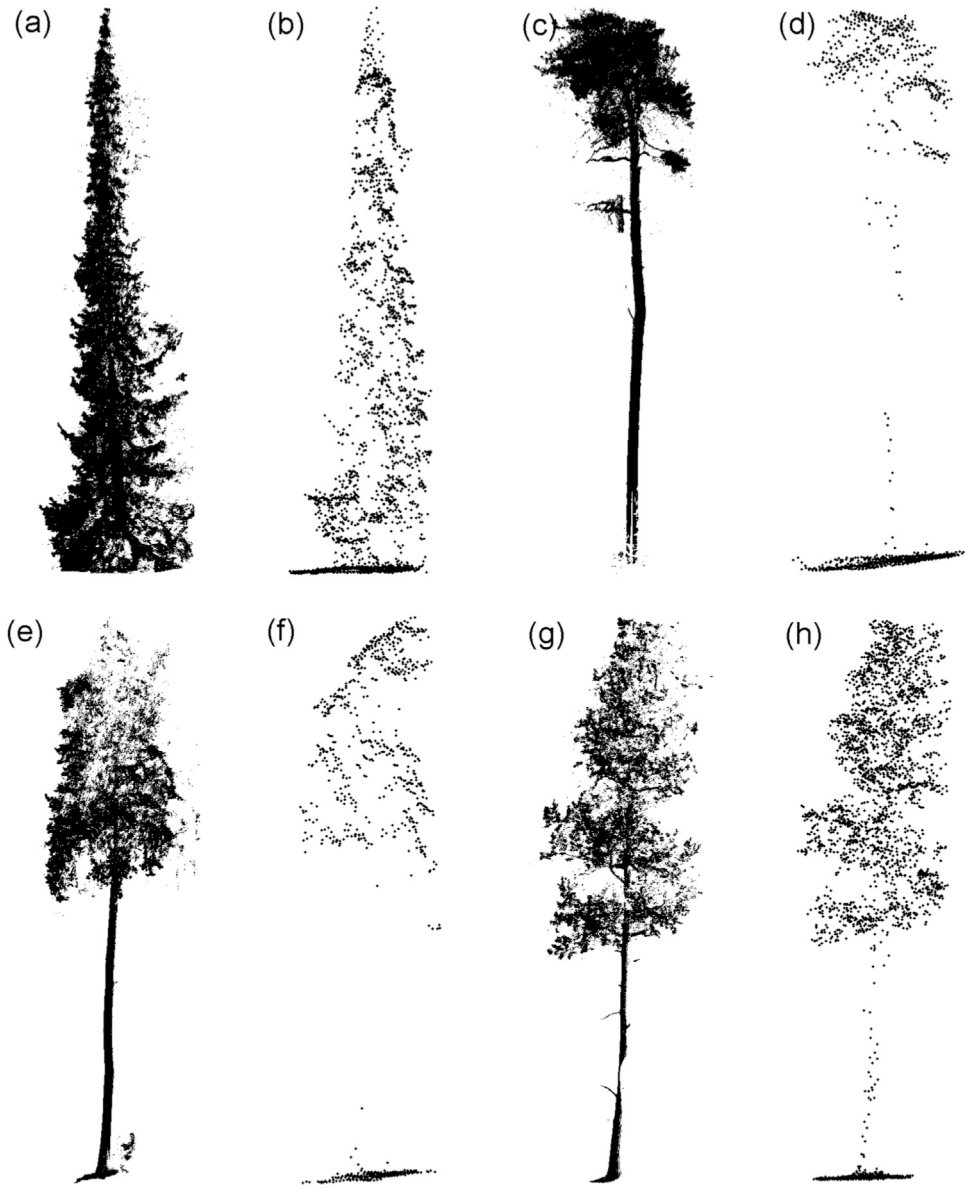
Luokittelukokeessa ilmalaserkeilausaineistona käytettiin 400 metrin korkeudesta Optech ALTM 3100 -skannerilla tehtyä keilausaineistoa, jonka pulssitiheys oli noin 10 pulssia/m^2. Kustakin pulssista saatiin 1-4 paluukaikua. Vertailuaineistona käytettiin Leica HDS6100 -maalaserkeilaimella hankittua korkearesoluutiodataa (kuva 3). Kukin otokseen valituista puista erotetiin pistepilvestä omaksi segmentikseen luokittelua varten.[16]

Linin ja Hyypän [16] menetelmässä tukivektorikonetta hyödynnetään yhden ulosjättävää ristiinvalidointia (leave-one-out cross-validation, LOOCV)-toimintamallilla, jossa kukin puu luokiteltiin käyttämällä otoksen muita puita opetusaineistona. Luokittelu tehtiin ensin yksitellen kuhunkin piirreparametriryhmään kuuluvien parametrien perusteella, minkä jälkeen haettiin optimaalinen piirreparametrien yhdistelmä, joka tuotti parhaan luokittelutarkkuuden. Kokeiden perusteella luokittelumalli todettiin toimivaksi. Menetelmän avulla voidaan pieneen otokseen perustuen nopeasti tunnistaa optimaaliset piirreparametrit puulajien tunnistamiseksi kulloinkin mielenkiinnon kohteena olevalla metsäalueella.

Puulajien tunnistamista tukivektorikonetta hyödyntäen ovat käsitelleet myös Dalponte ja Bruzzone [8]. Bosco della Fontanan luonnonsuojelualueella Italiassa tehdyssä kokeessa hyödynnettiin ilmalaserkeilausaineistoa ja keilauksen yhteydessä mitattua hyperspektriaineistoa, joka koostui 126 kaistasta kattaen aallonpituudet 400 - 990 nm. Mitattu alue kattoi noin 230 ha, josta

[1] Changin ja Linin LIBSVM-paketti ladattavissa osoitteesta `http://www.csie.ntu.edu.tw/~cjlin/libswm`, sivu ladattu 31.3.2016.

Kuva 3: Esimerkkikuvat tunnistettavista puulajeista, vasemmalla korkearesoluution-TLS ja oikealla ALS. Puulajit: (a) ja (b) kuusi, (c) ja (d) mänty, (e) ja (f) haapa sekä (g) ja (h) tammi. [16]



vertailuaineistoksi tehtiin maastotarkistuksena 550 puun otos. Luokittelussa hyödynnettiin laserkeilausaineistosta johdettuja korkeus- ja intensiteettitietoja sekä 40 hyperspektriaineiston kanavaa. Aineistosta pyrittiin erottamaan 20 eri puulajia sekä kolme muuta luokkaa.

Tukivektorikoneella tehdyssä luokittelussa saavutettiin merkittävästi tarkempia tuloksia kuin vertailumenetelmänä käytetyllä Gaussin suurimman todennäköisyyden menetelmällä. Laserkeilausaineistosta saatu korkeustieto oli tärkeässä roolissa samantyyppiset spektriset ominaisuudet omaavien puulajien erottelussa, ja kokeessa saavutettiinkin paikoin yli 90% luokittelutarkkuus. Tukivektorikoneen toimivuuden kannalta kriittisin tekijä on opetusaineiston ja varsinaisen luokiteltavan aineiston samankaltaisuus; mikäli opetusaineisto poikkeaa suuresti luokiteltavasta aineistosta, tukivektorinkoneen luokittelutarkkuus heikkenee voimakkaasti [8].

4.4 Esimerkki 2: Rakennetun ympäristön luokittelu

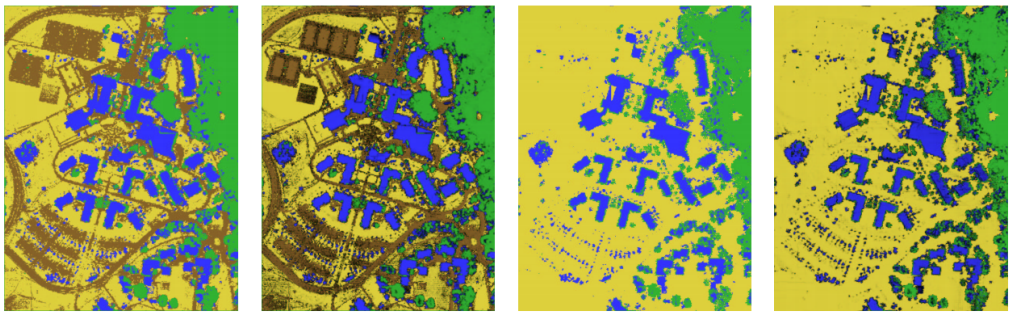
Rakennetussa ympäristössä tapahtuva luokittelu liittyy yleensä kaupunkimallinnukseen. Laserkeilaus mahdollistaa rakennetun ympäristön tarkan mallintamisen, ja sitä on hyödynnetty muun muassa rakennusten julkisivujen mallinnuksessa. Ilmalaserkeilauksen tarkkuus ei tosin mahdollista julkisivumallinnusta, mutta sillä voidaan kattaa nopeasti laajoja alueita, ja aineisto mahdollistaa karkeiden kaupunkimallien laskemisen. 3D-kaupunkimalleja voidaan hyödyntää muun muassa kaupunkisuunnittelussa, langattomien viestinten tukiasemien sijoittelussa ja melumallinnuksessa [30].

Tukivektorikoneen käyttöä rakennetun ympäristön kohteiden luokittelussa ovat käsitelleet Lodha ja muut [18], jotka pyrkivät ilmalaserkeilausaineiston ja ilmakuvien avulla luokittelemaan erilleen rakennukset, puuston, ruohikot ja tiet. Mallet ja muut [20] puolestaan käsittelevät rakennetun ympäristön luokittelua jatkuva-aaltoisen laserkeilausaineiston pohjalta, luokiteltavina kohteina rakennukset, puusto ja maanpinta.

Lodha ja muut [18] käyttivät tukivektorikonetta ilmalaserkeilausaineiston luokitteluun. Koealueena toimi Kalifornian yliopiston Santa Cruzin kampuksen ympäristöstä rajattu yli 20 neliökilometrin laajuinen alue, jossa tehdyn ilmalaserkeilauksen keskimääräinen pisteväli oli 0,26 metriä. Luokittelua varten aineisto interpoloitiin 0,5 metrin ruudukkoon käyttäen lähimmän naapurin menetelmää. Laserkeilausaineiston lisäksi käytettiin harmaasävyilmakuvia. Luokittelijana käytettiin LIBSVM-pakettiin [5] perustuvaa tukivektorikonetta.

Luokiteltavina piirteinä olivat normalisoitu korkeus, korkeuden vaihtelu, normaalivaihtelu, kaiun intensiteetti ja ilmavalokuvan intensiteetti. Normalisoidun korkeuden laskennassa mallista poistetaan maanpinnan vaikutus, joten vain pinnan yläpuolisella korkeudella on merkitystä. Korkeuden vaihtelu on laskettu 3 x 3 pikselin (2,25 m^2) ikkunaan ja koostuu korkeimman ja matalimman pikselin välisestä erotuksesta. Normaalivaihtelun laskennassa

Kuva 4: Ilmalaserkeilausaineiston SVM-luokittelukokeen tuloksia. 1. vasemmalta: neliluokkainen luokittelu rakennuksiin (sininen), puustoon (vihreä), teihin (ruskea) ja ruohikkoon (keltainen). 2. vasemmalta: neliluokkainen tulos, jossa sävy kuvaa luokittelun varmuutta (epävarmat kohteet tummempia). 3. vasemmalta: kolmiluokkainen tulos, jossa tiet yhdistetty ruohikkoon. 4. vasemmalta: kolmiluokkainen tulos, johon lisätty varmuustietoa kuvaava sävytys (epävarmat kohteet tummempia). [18]



lasketaan kullekin pisteeelle pintanormaali, ja 10 x 10 pikselin ruudukossa lasketaan näiden normaalien keskiarvo. Normaalivaihtelu kuvaa pinnan tasaisuutta. Laserkaiun intensiteetti kuvaa paluupulssin voimakkuutta ja ilmavalokuvan intesiteetti näkyvän valon heijastuksen voimakkuutta kohteesta. [18]

Kokeen tuloksena Lodha ja muut [18], esittävät, että tukivektorikoneella voidaan luokitella rakennetun ympäristön kohteet tarkasti ja toimintavarmasti (kuva 4). Menetelmässä on kuitenkin joitain heikkouksia, muun muassa kohina ja harhapisteet. Rakennusten mallinnusta hankaloittaa jonkin verran myös niiden nurkkien pyöristyminen. Spatiaalisen koherenssin huomioonottamisella tulosta olisi mahdollista parantaa, mutta sen liiallista käyttöä on myös syytä välttää, jotta erillisiä tai heikkoja piirteitä ei hävitetä.

Malletin ja muiden [20] hyödyntämä jatkuva-aaltoinen ilmalaserkeilaus odottaa vielä läpimurtoaan kaupallisissa ilmalaserkeilaussovelluksissa, mutta menetelmä on herättänyt mielenkiintoa erityisesti sen tuottaman mittausaineiston käsittelyn ja hallinnan osalta. Menetelmä tuottaa yksittäisten kaikujen sijaan kutakin mittauspistettä kohti yksiulotteisen aaltomuotoisen signaalin, josta kaiut lasketaan aineiston käsittelyvaiheessa. Koska signaalista voidaan erottaa runsaasti myös muuta tietoa, vaaditaan sen käsittelymenetelmiltä hyvää korkean dimensionaalisuuden sietoa. Tukivektorikone sopii ominaisuuksiltaan hyvin tämän tyyppisten ainestojen käsittelyyn.

Koealueena Mallet ja muut käyttivät noin 1,1 km^2 aluetta Biberachissa, Baden-Wurttenbergissä, Saksassa. Kesäaikaan mitatun aineiston resoluutio oli noin 5 pistettä / m^2. Aineisto koostui noin 2,2 miljoonasta aaltomuodosta, jotka kattoivat erityyppisten rakennettujen alueiden ohella metsämaastoa. Aaltomuodoista laskettiin laserkaikuja vastaavat pisteet ja jatkokäsittely teh-

tiin tämän pistepilven pohjalta. Varsinaisessa luokitteluvaiheessa käsiteltävä aineisto oli siis saman tyyppistä kuin pulssilaserkeilainten tuottama data. Ilmalaserkeilausaineiston lisäksi käytössä oli alueen kattava ortoilmakuva, jota käytettiin tuloksen tarkastelussa, ei varsinaisessa luokittelussa.

Luokkakohtaisesti valittiin satunnaisotannalla 100 näytettä, joita käytettiin opetusjoukkona, ja jotka edustivat 0,01% luokiteltavasta aineistosta. Lisäksi tuloksen ristiinvalidoinnissa käytettiin 0,05% kokoista otosta.

Luokittelussa keskityttiin rakennuksiin, puustoon ja maanpintaan. Piirreparametrejä oli 27, joista 19 oli pistepilven geometrisiin ominaisuuksiin ja kaikujen voimakkuuteen liittyviä ja loput kahdeksan oli suoraan aaltomuodosta johdettuja.

Tärkeimpinä luokkia erottavina piirteinä pidettiin aaltomuodosta johdettua amplitudia, kaiun halkaisijaa ja takaisinsirontavakiota. Sen sijaan geometriaa kuvaavan tasaisuuden ja vinouden todettiin olevan vähiten merkitseviä piirteitä. Ydinfunktion valinnalla todettiin olevan jonkin verran vaikutusta lopputulokseen, mutta koska luokkia oli vain kolme, ei sen katsottu olevan merkittävässä roolissa. Sen sijaan amplitudikorjauksen ja mittausaineiston kalibraation merkitystä korostettiin erottavien piirteiden tunnistamisessa. [20]

Saavutettu luokittelutarkkuus vaihteli 80% ja 90% välillä, minkä perusteella voidaan todeta, että tukivektorikone ei tehnyt ylisovitusta, vaan yleisti hyvin, ja oli asianmukaisesti opetettu [20].

4.5 Esimerkki 3. Maanvyöryriskin kartoitus

Maanvyöryherkillä alueilla vöyrymäriskialueiden tunnistaminen on tärkeää sekä maankäytön suunnittelun että pelastuspalvelun toimintojen organisoimiseksi. Riittävän jyrkkä topografia yhdistettynä runsassateiseen ilmastoon ja epävakaaseen kallioperään luovat otolliset mahdollisuudet maanvyörymille, jotka voivat pahimmillaan aiheuttaa merkittävää aineellista ja inhimillistä tuhoa.

Jebur ja muut [14] käsittelevät maanvyörymäriskin ennakointia ilmalaserkeilausaineiston, geologisen ja maankäyttöä koskevan tiedon perusteella. Tavoitteena oli löytää parametrien yhdistelmä, jolla riskiä voitaisiin luotettavasti ennakoida. Koealueena toimi Bukit Antarabangsan alue Ulku Klangissa, Malesiassa. Alueella on korkea maanvöyryriski liittyen mm. eroosioon, suuriin sademääriin ja epävakaaseen geologiaan. Opetuskohteet koostuivat 31 tapahtuneesta maanvyörystä.

Maanvyörymäriskikartoituksessa käytettiin kahta rinnakkaista aineistoa. Ensimmäinen koostui ilmalaserkeilausaineistosta ja siitä johdetuista parametreistä: korkeus merenpinnasta, rinteen jyrkkyys, rinteen suunta, kaarevuus, virran voimakkuusindeksi (Stream Power Index, SPI), topografinen märkyysindeksi (topographic Wetness Index, TWI), topografinen karkeusindeksi (Topographic Roughness Index, TRI) sekä sedimentin kuljetusindeksi

(Sediment Transport Index). Toinen aineisto koostui ensimmäisen parametreista, joita täydennettiin geologisella tiedolla maalajeista ja kallioperästä, maankäyttöä ja kasvillisuutta koskevalla LULC-tiedolla (Land Use / Land Cover) sekä etäisyystiedolla jokiin ja teihin.

Käytetyt mallinnusmenetelmät olivat weight of evidence (WoE), logistinen regressio (LR) sekä tukivektorikone. WoE on kaksimuuttujainen tilastollinen menetelmä ja logistinen regressio monimuuttujainen tilastollinen menetelmä. Tulosten arviointiin käytettiin Area Under Curve -menetelmää (AUC).

Menetelmistä puhtaasti ilmalaserkeilausaineistosta johdettuja muuttujia käsiteltäessä LR ja SVM tuottivat parhaan tuloksen, saavuttaen 86% ja 84% ennustustarkkuuden, WoE:n jäädessä 59% tarkkuuteen. Geologisella ja maankäytöllisellä tiedolla täydennettyä jälkimmäistä aineistoa käsiteltäessä tulokset jäivät LR:n ja SVM:n osalta heikommiksi, 66% ja 69%, mutta WoE tuotti paremman tuloksen (65%).

Kokeeseen perustuen voidaan todeta, että sopivan mallinnusmenetelmän avulla (LR tai SVM) maanvöyrymäriskiä voidaan kohtuullisen luotettavasti arvioida pelkästään ilmalaserkeilausaineistosta johdettujen muuttujien pohjalta. Pientä epätarkkuutta tulokseen jää, koska maanvyöryihin liittyy aina vaikeasti arvioitavia paikallisia muuttujia, jotka ovat ollee myötävaikuttamassa niiden syntyyn. Lisäksi vöyryriskikartoitusten tekijät ovat käytetyistä menetelmistä johtuen useimmiten paikkatietojärjestelmäasiantuntijoita, eivätkä välttämättä omaa kohdealueen maa- ja kallioperään liittyvää asiantuntemusta, mikä heikentää tuloksen tarkkuutta. [14]

5 Yhteenveto ja pohdintaa

Laserkeilaus on mittausmenetelmä, jolla kolmiulotteisista kohteista voidaan luoda tarkkoja malleja. Menetelmä tuottaa suuria määriä mittausdataa, jolla on kuitenkin rajallinen määrä piirreavaruuksia. Mittausdatasta on kuitenkin mahdollista johtaa lukuisia uusia piirreparametreja, mikä nopeasti kasvattaa luokittelussa hyödynnettävien piirreparametrien määrää. Laserkeilausaineistoon voidaan myös yhdistää muilla sensoreilla hankittua dataa, kuten esimerkiksi ortokuvia tai hyperspektridataa, jotka myös kasvattavat piirreparametrien määrää. Erityissovelluksissa, kuten kasvillisuuden luokitteluun liittyvissä sovelluksissa piirreparametrien määrä kasvaa helposti niin suureksi, että se hankaloittaa tavanomaisten luokittelumenetelmien käyttöä.

Tukivektorikone on suhteellisen pitkästä iästään huolimatta edelleen varsin tuore menetelmä laserkeilausaineiston luokittelussa. Vakiintuneisiin menetelmiin verrattuna sillä on joitain hyviä puolia, kuten opetusprosessin keveys ja korkean dimensionaalisuuden sieto, mikä etenkin piirreparametrien määrän kasvaessa helpottaa opetusprosessia. Tukivektorikoneen avulla on mahdollista luokitella runsaasti eri piirreparametreja sisältäviä aineistoja tar-

kasti ja hyvällä varmuudella. Menetelmän hyvä yleistyskyky nopeuttaa myös sen toimintaa. Menetelmän huonoina puolina mainitaan sen hitaus varsinaisen aineiston luokittelussa. Se ei myöskään ota huomioon vierekkyyttä, mikä joissain sovelluksissa saattaa aiheuttaa väärin luokiteltuja harhapisteitä.

Pisimmälle tukivektorikoneen hyödyntäminen laserkeilausaineiston luokittelussa on viety metsätalouden ja -ekologian sovelluksissa, joissa se on havaittu toimivaksi luokittelumenetelmäksi. Puulajien tunnistuksessa menetelmää on käytetty menestyksekkäästi sekä puhtaasti ilmalaserkeilausaineistoon pohjautuvien että siihen yhdistettyjen muilla menetelmillä saatujen piirreparametrien avulla tapahtuvassa luokittelussa.

Tukivektorikonetta on hyödynnetty myös rakennetun ympäristön kohteiden luokittelussa. Tavoitteena on ollut automatisoida ilmalaserkeilausaineiston käsittelyä niin, että siitä pystyttäisiin tunnistamaan muun muassa rakennukset, kasvillisuus ja maanpinta automaattisesti. Tätä tietoa voidaan hyödyntää kaupunkimallinnuksessa ja kaupunkisuunnittelussa.

Tukivektorikonetta on myös hyödynnetty luonnonmuodostelmiin liittyvässä riskikartoituksessa. Menetelmällä on haettu optimaalisia muuttujia potentiaalisien maanvyörymäriskialueiden kartoitukseen ilmalaserkeilausaineiston ja muun geologisen ja maankäyttöön liittyvän tiedon pohjalta. Kokeissa havaittiin, että riskiarvio on mahdollista tehdä suhteellisen luotettavasti ilman geologista ja maankäyttöön liityvää tietoa, puhtaasti ilmalaserkeilausaineiston pohjalta, mikä mahdollistaa riskianalyysin tekemisen huomattavisti nopeammin ja helpommin kuin kaikki aineistot yhdistämällä.

Tukivektorikoneella on aineistojen luokittelun ohella sovelluksia muun muassa laserkeilausaineistojen laadun parantamisessa. Menetelmän käyttöä laserkeilausaineiston kohinan poistoon on tutkittu, ja siitä on saatu hyviä tuloksia.

Tukivektorikone on menetelmä, jonka yleistyminen on tapahtunut varsin hitaasti. Sen juuret ovat 1960-luvulla tehdyissä tutkimuksissa ja menetelmä on varsinaisesti esitelty vasta 1980-luvulla, mutta sen yleistyminen on tapahtunut vasta 2000-luvulla. Parhaiten se on hyväksytty runsaasti piirreparametreja sisältävien ainestojen käsittelyssä, mutta myös laserkeilaukseen liittyviä sovelluksia on esitelty useita. Joka paikan menetelmää siitä ei kuitenkaan ole vielä tullut. Merkittävimmät esteet tukivektorikoneen hyödyntämisessä laserkeilausaineistoihin liittyen liittyvät sen sovellusalaan, sillä tukivektorikoneen käyttö vaatii ymmärrystä koneoppimisesta ja sen lisäksi tietojenkäsittelytieteestä.

Ilmalaserkeilausaineistojen käsittelyn tulevaisuus saattaa kuitenkin avata tukivektorikoneelle uusia sovelluksia. Jatkuva-aaltoisen laserkeilauksen yleistyessä sen tuottamaan mittausdataan liittyvät haasteet ja siitä erotettavien piirreparametrien määrän kasvu saattaa nostaa mielenkiintoa tukivektorikonetta kohtaan. Samoin saattaa käydä, mikäli lasertekniikassa saavutetaan edistystä moni- tai hyperkanavaisissa lasereissa, mikä myös kasvattaisi laserkeilauksen yhteydessä tallennettavien piirreparametrien määrää. Nämä kehi-

tysaskeleet tuovat mukanaan myös haasteita liittyen tiedon tallennukseen ja käsittelyyn, mikä jo nyt tapahtuu nykyaikaisten tietokoneiden käsittely- ja tallennuskapasiteetin rajoilla.

Viitteet

[1] Alexander S. Antonarakis, Keith S. Richards, and James Brasington. Object-based land cover classification using airborne LiDAR. *Remote Sensing of Environment*, 112(6):2988 – 2998, 2008.

[2] Peter Axelsson. Processing of laser scanner data-algorithms and applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2-3):138 – 147, 1999.

[3] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121 – 167, 1998.

[4] Colin Campbell. Kernel methods: a survey of current techniques. *Neurocomputing*, 48(1 - 4):63 – 84, 2002.

[5] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intellicent Systems and Technology*, 2(3):27:1 – 27:27, May 2011.

[6] Yunhao Chen, Wei Su, Jing Li, and Zhongping Sun. Hierarchical object oriented classification using very high resolution imagery and LIDAR data over urban areas. *Advances in Space Research*, 43(7):1101 – 1110, 2009.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273 – 297, 1995.

[8] Michele Dalponte, Lorenzo Bruzzone, and Damiano Gianelle. Fusion of hyperspectral and lidar remote sensing data for classification of complex forest areas. *IEEE Transactions on Geoscience and Remote Sensing*, 46(5):1416 – 1427, May 2008.

[9] Mathieu Fauvel, Jocelyn Chanussot, and Jón Atli Benediktsson. Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas. *EURASIP Journal on Advances in Signal Processing*, 2009:11:1 – 11:14, January 2009.

[10] Norbert Haala, Michael Peter, Jens Kremer, and Graham Hunter. Mobile LiDAR mapping for 3d point cloud collection in urban areas – a performance test. *The international archives of the photogrammetry, remote sensing and spatial information sciences*, 37:1119 – 1127, 2008.

[11] Kyle A. Hartfield, Katheryn I. Landau, and Willem J. D. van Leeuwen. Fusion of high resolution aerial multispectral and LiDAR data: Land cover in the context of urban mosquito habitat. *Remote Sensing*, 3(11):2364, 2011.

[12] Xin Huang, Liangpei Zhang, and Wei Gong. Information fusion of aerial images and lidar data in urban areas: vector-stacking, re-classification and post-processing approaches. *International Journal of Remote Sensing*, 32(1):69 – 84, 2011.

[13] Jungho Im, John R. Jensen, and Michael E. Hodgson. Object-based land cover classification using high-posting-density LiDAR data. *GIScience & Remote Sensing*, 45(2):209–228, 2008.

[14] Mustafa Neamah Jebur, Biswajeet Pradhan, and Mahyat Shafapour Tehrany. Optimization of landslide conditioning factors using very high-resolution airborne laser scanning (lidar) data at catchment scale. *Remote Sensing of Environment*, 152:150 – 165, 2014.

[15] Sanna Kaasalainen, Antero Kukko, Tomi Lindroos, Paula Litkey, Harri Kaartinen, Juha Hyyppa, and Eero Ahokas. Brightness measurements and calibration with airborne and terrestrial laser scanners. *IEEE Transactions on Geoscience and Remote Sensing*, 46(2):528 – 534, Feb 2008.

[16] Yi Lin and Juha Hyyppä. A comprehensive but efficient framework of proposing and validating feature parameters from airborne LiDAR data for tree species classification. *International Journal of Applied Earth Observation and Geoinformation*, 46:45 – 55, 2016.

[17] Yi Lin, Juha Hyyppä, and Anttoni Jaakkola. Mini-uav-borne lidar for fine-scale mapping. *Geoscience and Remote Sensing Letters, IEEE*, 8(3):426 – 430, May 2011.

[18] Suresh K. Lodha, Edward J. Kreps, David P. Helmbold, and Darren N. Fitzpatrick. Aerial lidar data classification using support vector machines (svm). In *3DPVT*, pages 567 – 574, 2006.

[19] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing*, 28(5):823 – 870, 2007.

[20] Clément Mallet, Frédéric Bretar, Michel Roux, Uwe Soergel, and Christian Heipke. Relevance assessment of full-waveform lidar data for urban area classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(6, Supplement):S71 – S84, 2011. Advances in LIDAR Data Processing and Applications.

[21] Farid Melgani and Lorenzo Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on Geoscience and Remote Sensing*, 42(8):1778 – 1790, Aug 2004.

[22] Giorgos Mountrakis, Jungho Im, and Caesar Ogole. Support vector machines in remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3):247 – 259, 2011.

[23] Erik Næsset, Terje Gobakken, Johan Holmgren, Hannu Hyyppä, Juha Hyyppä, Matti Maltamo, Mats Nilsson, Håkan Olsson, Åsa Persson, and Ulf Söderman. Laser scanning of forest resources: the nordic experience. *Scandinavian Journal of Forest Research*, 19(6):482 – 499, 2004.

[24] Mahesh Pal and Paul M. Mather. Support vector machines for classification in remote sensing. *International Journal of Remote Sensing*, 26(5):1007 – 1011, 2005.

[25] Gordon Petrie and Charles K Toth. Terrestrial laser scanners. *Topographic Laser Ranging and Scanning Principles and Processing*, pages 87 – 128, 2009.

[26] Norbert Pfeifer and Christian Briese. Geometrical aspects of airborne laser scanning and terrestrial laser scanning. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(3/W52):311 – 319, 2007.

[27] George Sithole and George Vosselman. Experimental comparison of filter algorithms for bare-earth extraction from airborne laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(1 - 2):85 – 101, 2004. Advanced Techniques for Analysis of Geo-spatial Data.

[28] Arttu Soininen and Hannu Korpela. Processing of airborne laser data and images–versatile products through skilled processing. *GIS development magazine*, 10, 2007.

[29] Bing-Yu Sun, De-Shuang Huang, and Hai-Tao Fang. Lidar signal denoising using least-squares support vector machine. *IEEE Signal Processing Letters*, 12(2):101 – 104, Feb 2005.

[30] Aloysius Wehr and Uwe Lohr. Airborne laser scanning - an introduction and overview. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2 – 3):68 – 82, 1999.

[31] Wai Yeung Yan, Ahmed Shaker, and Nagwa El-Ashmawy. Urban land cover classification using airborne LiDAR data: A review. *Remote Sensing of Environment*, 158:295 – 310, 2015.

[32] Guobin Zhu and Dan G. Blumberg. Classification using ASTER data and SVM algorithms: The case study of Beer Sheva, Israel. *Remote Sensing of Environment*, 80(2):233 – 240, 2002.

Sovelluskerroksen tietojen poiminta ja välitys tietoverkkoliikenteestä IPFIX-protokollan avulla

Lasse Tuominen

Tiivistelmä.
Tietoverkkoliikenteestä voidaan poimia ja välittää erilaisia tietoja eteenpäin analysoitavaksi. Tietoja voidaan poimia esimerkiksi erilaisilta antureilta tai verkkolaitteilta kuten reitittimiltä, palomuureilta ja kuormanjakajilta. Tässä tutkielmassa tutkitaan mahdollisuuksia käyttää IPFIX-protokollaa sovelluskerroksen tietojen poimintaan ja välittämiseen analysointia varten. Esimerkkeinä sovelluskerroksen protokollista ovat HTTP, HTTPS ja SIP.

Avainsanat ja -sanonnat: Deep Packet Inspection, televalvonta, tietoverkko, tietovirta.

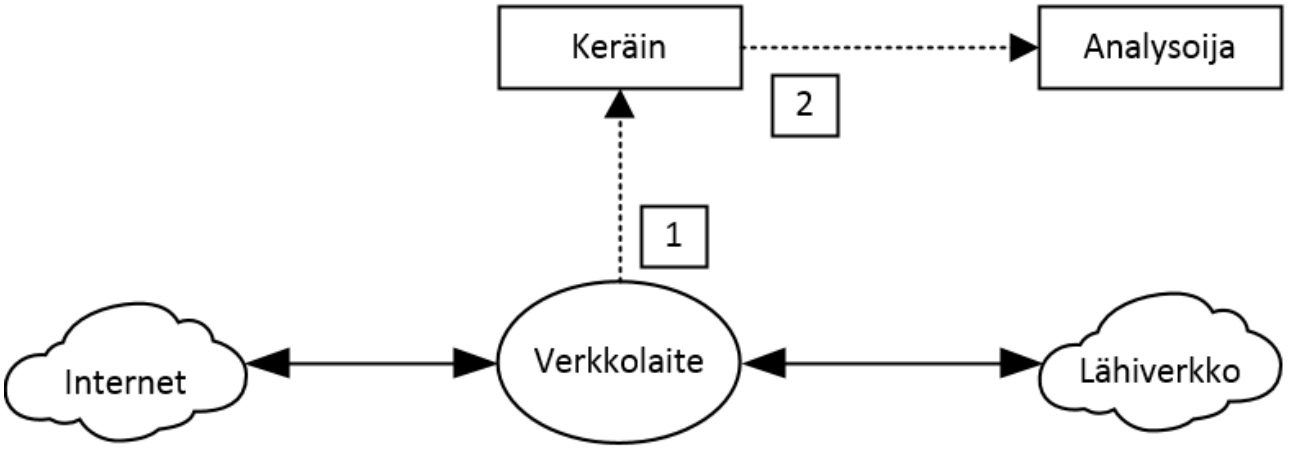
1. Johdanto

Tietoverkkoliikenteen valvontaan ja analysointiin on olemassa useita eri tekniikoita. Jotkin näistä tekniikoista hyödyntävät kaupallisia ohjelmistoja tai erikoisvalmisteisia laitteita ja voivat vaatia paljon laskentatehoa sekä tallennuskapasiteettia. Valvontaan on kaksi erilaista lähestymistapaa. Ensimmäinen vaihtoehto on *Deep Packet Inspection* (DPI) eli pakettien syväluotaus, joka antaa näkyvyyden sovelluskerroksen tietoihin. Pakettien syväluotauksen haittapuolena pidetään pakettien yksityiskohtaisen tarkastelun vaatimaa korkeaa laskentatehoa [Velan and Čeleda 2013]. Toinen lähestymistapa on IP-liikenteen virtojen seuraaminen, joka toteutetaan tarkastelemalla vain pakettien otsikoita. Virtojen seuraamisen haittapuolena on, että näkyvyys rajoittuu korkeintaan kuljetuskerroksen protokolliin [Velan and Čeleda 2013]. Nämä kaksi lähestymistapaa voidaan kuitenkin yhdistää esimerkiksi käyttämällä IPFIX-protokollaa.

Tässä tutkielmassa esitellään ja sovelletaan IPFIX-protokollan käyttämistä sovelluskerroksen tietojen tutkimiseen. IPFIX eli *IP Flow Information Export* on standardoitu protokolla tietoliikennevirran tietojen välittämiseen, vastaanottamiseen ja esittämiseen tietoliikenneverkossa. Virta muodostuu joukoista paketteja, jotka kulkevat tietyn tarkkailupisteen läpi tiettynä ajanhetkenä [RFC 7011 2013]. IPFIX-protokollan toiminta perustuu myöhemmin esiteltäviin mallineisiin, datatietueisiin ja informaatioelementteihin. Vaikka tutkielmassa esitellään joitakin asioita verkkotekniikasta ja sovelluskerroksen protokollista, on aiempi tuntemus verkkotekniikasta suotavaa aihepiirin ja käsiteltyjen teemojen ymmärtämiseksi.

Kuvassa 1 esitellään tyypillinen IPFIX-arkkitehtuuri. Tässä arkkitehtuurissa jokin verkkolaite on tarkkailupisteenä Internetin ja lähiverkon väliselle liikenteelle. Verkkolaitteelta

lähetetään tiedot keräimelle (1) ja keräin lähettää tiedot eteenpäin analysoitavaksi (2). Standardin mukaan tiedot voidaan lähettää eteenpäin käyttäen jotakin seuraavista kuljetuskerroksen protokollista: *Stream Control Transmission Protocol* (SCTP), *Transmission Control Protocol* (TCP) tai *User Datagram Protocol* (UDP).



Kuva 1. IPFIX-arkkitehtuuri.

Seuraavassa luvussa kuvataan, miten tietoliikennevirtojen sisältöjen tiedot kuvataan IPFIX-protokollassa. Lisäksi tarkastellaan tarkemmin informaatioelementtejä ja varsinkin yrityskohtaisia informaatioelementtejä. Kolmannessa luvussa kuvataan järjestelmän arkkitehtuuria eli erilaisia tietoja lähettäviä ja tietoja vastaanottavia komponentteja. Näihin lukeutuvat erilaiset verkkolaitteet, sovellukset ja ohjelmistot. Luvussa 4 käydään läpi sovelluskerroksen tietojen poimintaa tietoliikennevirroista ja niiden käyttötapauksia. Viides luku sisältää yhteenvedon, tämän hetkisten ongelmakohtien ja tulevaisuuden mahdollisuuksien pohdintaa IPFIX-protokollan käytöstä tietoverkkoliikenteen seurannassa.

2. Tietojen esitystapa

2.1. Datatietueet ja mallineet

IPFIX-protokolla esittää tiedot datatietueina, jotka määritellään lähetettävissä mallineissa ennen tietueiden lähettämistä. Mallineissa määritellään datatietueiden kunkin informaatioelementin nimi, tunniste ja sisällön maksimipituus. Datatietueissa olevat informaatioelementit koostuvat pakollisista kentistä, joita ovat nimi, tunniste, kuvaus, tietotyyppi ja tila. Näiden lisäksi informaatioelementeillä ovat mahdollisia kenttinä tietotyypin merkitys, mittayksikkö, arvoalue, referenssi, versionumero ja päivämäärä. Myöhemmin mainittavissa yrityskohtaisissa informaatioelementeissä on myös yrityksen tunniste eli *Private Enterpise*

Number (PEN) [IANA 2016a]. Tietoliikennevirran tietojen lähetyksessä ja vastaanotossa kuitenkin käytetään vain nimeä, tunnistetta ja tietotyyppiä.

Tietotyyppi määrittelee, mitä tietoa informaatioelementti sisältää. Mahdollisia tyyppejä ovat esimerkiksi numeeriset arvot, merkkijonot, totuusarvot, MAC-osoitteet, IP-osoitteet tai aikaleima eri muodoissa. Informaatioelementit jakaantuvat IPFIX-protokollassa kahteen eri ryhmään: *Internet Assigned Numbers Authority* -järjestön (IANA) määrittelemiin ja itse määriteltäviin yrityskohtaisiin eli *enterprise-specific* informaatioelementteihin.

Kuvassa 2 esitellään malline, jossa on sekä IANA:n että Tampereen yliopiston määrittelemiä informaatioelementtejä. Kuvassa esitellään tarkemmin kentät 4 (*IP_DST_ADDR*) ja 11 (*1001 [pen: University of Tampere]*).

```
▷ Frame 1: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits)
▷ Ethernet II, Src: Vmware_ab:21:ae (00:50:56:ab:21:ae), Dst: Vmware_b6:d5:ee (00:50:56:b6:d5:ee)
▷ Internet Protocol Version 4, Src: 172.16.30.5, Dst: 172.16.30.10
▷ User Datagram Protocol, Src Port: 17699 (17699), Dst Port: 4739 (4739)
◢ Cisco NetFlow/IPFIX
    Version: 10
    Length: 84
  ▷ Timestamp: May 24, 2016 14:24:28.000000000 FLE Daylight Time
    FlowSequence: 0
    Observation Domain Id: 171053066
  ◢ Set 1 [id=2] (Data Template): 256
      FlowSet Id: Data Template (V10 [IPFIX]) (2)
      FlowSet Length: 68
    ◢ Template (Id = 256, Count = 12)
        Template Id: 256
        Field Count: 12
      ▷ Field (1/12): flowStartMilliseconds
      ▷ Field (2/12): IP_SRC_ADDR
      ▷ Field (3/12): TCP_SRC_PORT
      ◢ Field (4/12): IP_DST_ADDR
          0... .... .... .... = Pen provided: No
          .000 0000 0000 1100 = Type: IP_DST_ADDR (12)
          Length: 4
      ▷ Field (5/12): TCP_DST_PORT
      ▷ Field (6/12): flowDurationMilliseconds
      ▷ Field (7/12): PROTOCOL
      ▷ Field (8/12): BYTES
      ▷ Field (9/12): PKTS
      ▷ Field (10/12): 1000 [pen: University of Tampere]
      ◢ Field (11/12): 1001 [pen: University of Tampere]
          1... .... .... .... = Pen provided: Yes
          .000 0011 1110 1001 = Type: 1001 [pen: University of Tampere]
          Length: 65535 [i.e.: "Variable Length"]
          PEN: University of Tampere (25182)
      ▷ Field (12/12): 1002 [pen: University of Tampere]
```

Kuva 2. IPFIX-protokollan malline, jossa sekä IANA:n että Tampereen yliopiston määrittelemiä informaatioelementtejä.

2.2. IANA:n määrittelemät informaatioelementit

IANA:n määrittelemät informaatioelementit sisältävät yleisiä tietoja, kuten virran alkamis- ja loppumisajankohta, kokonaiskesto, lukumäärä ja käytetty protokolla. Näiden lisäksi on useita verkko- ja kuljetuskerroksen sekä joitakin siirtokerroksen tietoja sisältäviä informaatioelementtejä [IANA 2016b]. Monissa ratkaisuissa tietoverkkoliikenteestä saatavien tietojen laajuuteen riittävät IANA:n määrittelemät informaatioelementit.

Esimerkkejä näistä tiedoista ovat aikaisemmin mainittujen yleisien tietojen lisäksi lähde- ja kohdeosoitteet, lähde- ja kohdeportit sekä pakettien ja oktettien lukumäärät. Mikäli kuitenkin halutaan tutkia sovelluskerroksen tietoja, on IANA määritellyt informaatioelementtejä vain esimerkiksi BGP-reititykseen. Muihin käyttötapauksiin tarvitaan yrityskohtaisia informaatioelementtejä, joista käytetään joissakin yhteyksissä myös *extended flow records* -termiä.

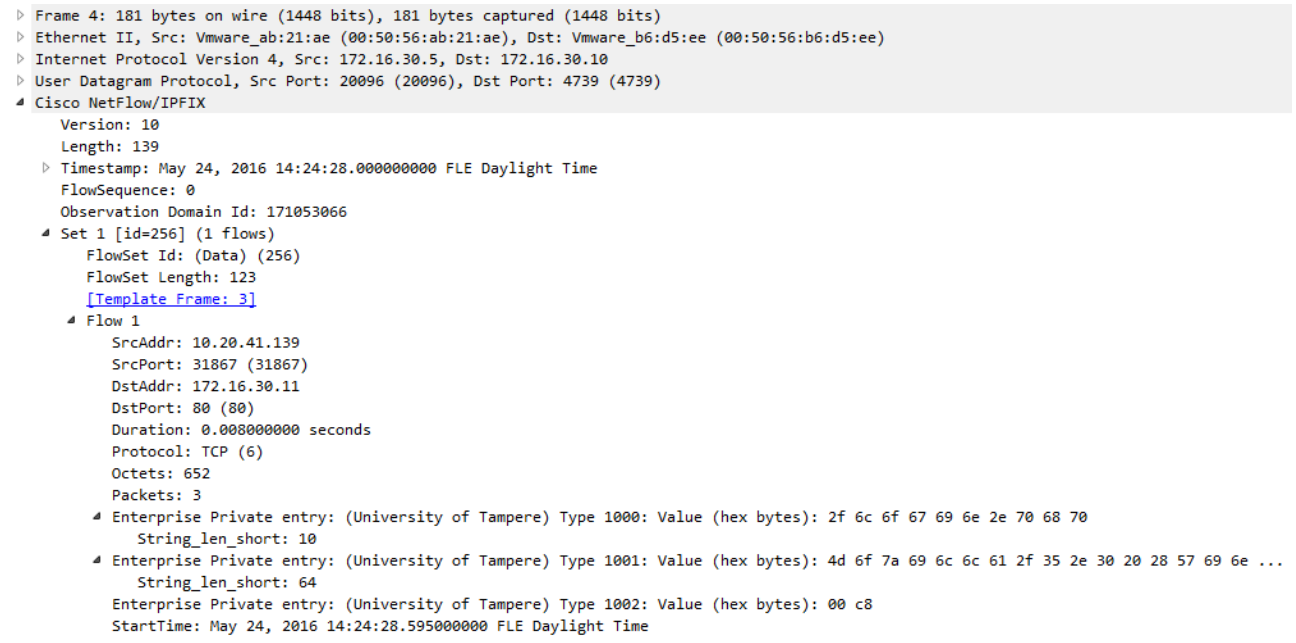
2.3. Yrityskohtaiset informaatioelementit

Yrityskohtaiset (e*nterprise-specific*) informaatioelementit eroavat IANA:n määrittelemistä informaatioelementeistä kahdella olennaisella tavalla: niitä voi määritellä itse ja ne sisältävät aikaisemmin mainitun PEN:in eli yrityksen tunnisteen. Yrityksen tunnisteen käyttäminen on pakollinen määriteltävä arvo. Tunnisteen rekisteröinti yritykselle on kuitenkin tarpeellista vain tapauksissa, joissa informaatioelementit tuodaan julkisiksi tai informaatioelementtejä käytetään yrityksen ulkopuolella [RFC 7012 2013].

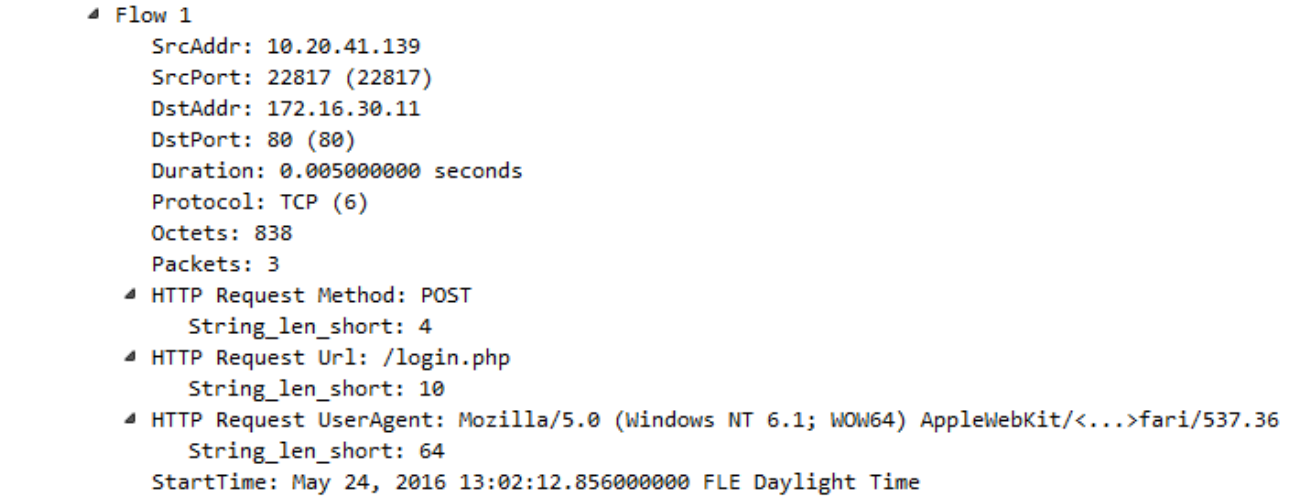
Informaatioelementtien määritteleminen itse tarkoittaa käytännössä sitä, että kohdssa 3.1 ja 3.2 esiteltäviin lähettäjään ja vastaanottajaan tulee määritellä informaatioelementin nimi, tunniste, tietotyyppi, pituus ja yrityksen tunniste. Tämän lisäksi lähettäjällä pitää myös määritellä lähetettävä malline. Yrityskohtaisten informaatioelementtien määritteleminen ei rajoitu pelkästään sovelluskerroksen tietoihin vaan informaatioelementtejä voidaan määritellä vapaasti.

Kuvassa 3 esitellään kuvan 2 mukaisen mallineen määrittelemät datatietueet, jotka sisältävät tiedot yhdestä tietoliikennevirrasta. Kolme viimeistä kenttää ovat yrityskohtaisia informaatioelementtejä. Tässä tapauksessa ne ovat Tampereen yliopiston PEN:illä luotuja informaatioelementtejä, jotka sisältävät seuraavat tiedot HTTP-liikenteen otsikoista: pyynnön URI (*1000*), pyynnön asiakasohjelma (*1001*) ja pyynnön vastauskoodi (*1002*).

Tiedot näytetään kuvassa heksadesimaalilukuina, koska käytetty Wireshark-ohjelma ei tunnista kyseisiä informaatioelementtejä toisin kuin kuvassa 4 näytetyt Citrix Netscalerin käyttämät informaatioelementit: *HTTP Request Method*, *HTTP Request Url* ja *HTTP Request UserAgent* [Wireshark 2016].



Kuva 3. IPFIX-protokollan datatietueet kuvan 2 mallineen mukaan.



Kuva 4. IPFIX-protokollan datatietueita.

3. Järjestelmäarkkitehtuuri

Luvussa 1 esitetyn kuvan 1 osoittamalla tavalla järjestelmä koostuu jonkinlaisesta verkkolaitteesta eli tietojen lähettäjästä ja keräimestä eli tietojen vastaanottajasta. Lisäksi järjestelmässä voi olla erillinen analysoija, jonka avulla voidaan nimenmukaisesti analysoida ja visualisoida tietoja tai luoda erilaisia raportteja tietojen perusteella.

3.1. Tietojen lähettäjä

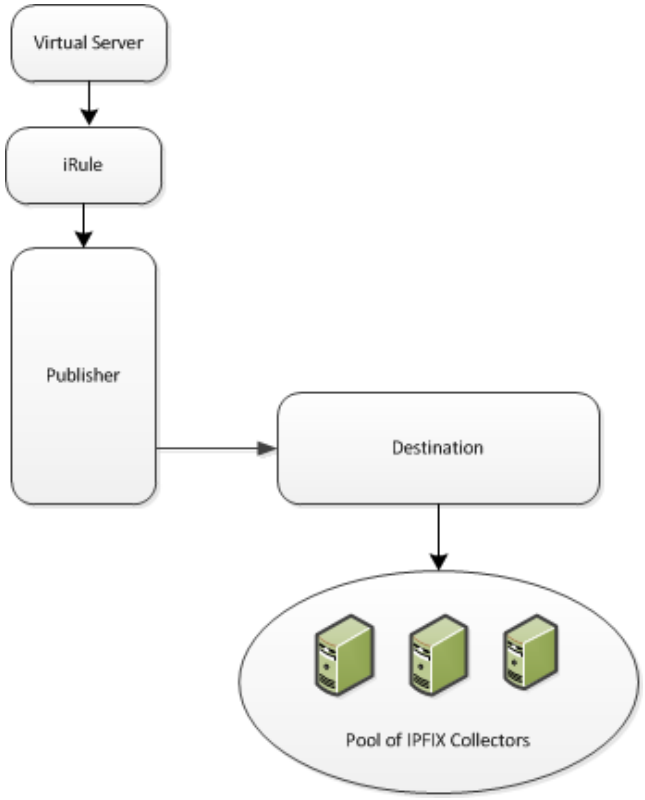
Analysoitavan tiedon lähettäjänä toimii jokin tietoliikenneverkossa sijaitseva anturi tai verkkolaite, kuten esimerkiksi palomuuri, reititin tai kuormanjakaja. Antureiden tapauksessa tietoverkkoliikenne monesti peilataan anturille tai käytetään jotain muuta tekniikkaa, jolla liikenne saadaan näkyville anturiin. Palomuureissa, reitittimissä ja kuormanjakajissa tietoverkkoliikenne kulkee laitteiden läpi ja tätä kautta saadaan suoraan näkyvyys liikenteeseen.

Palomuurit toimivat pääsääntöisesti kuljetuskerroksessa, joka rajoittaa palomuureista saatavia tietoja. Tällä hetkellä markkinoilla on myös niin kutsuttuja seuraavan sukupolven palomuureja *(Next-Generation Firewall*), joilla on näkyvyys sovelluskerrokseen asti. Reitittimet toimivat korkeintaan verkkokerroksessa, joten niiltä saatavat tiedot ovat vielä rajoitetumpia kuin palomuureista. Palomuurien ja joidenkin reitittimien tapauksissa tietoverkkoliikenteen analysointi tapahtuu usein käyttämällä protokollia, jotka perustuvat erillisten näytteiden ottoon. Tällaisia ovat esimerkiksi jFlow ja sFlow.

Käsitellään seuraavaksi kahden eri kuormanjakajan kyvykkyydet poimia tietoja tietoverkkoliikenteestä ja lähettää ne eteenpäin IPFIX-protokollan avulla. Kuormanjakajiin usein myös päättyy salattu yhteys esimerkiksi HTTPS-liikenteen tapauksessa.

Aloitetaan F5 Networksin BIG-IP Local Traffic Manager -tuotteella. Se käyttää laitteen säännöstöjä (*iRules*) lukeakseen liikennettä ja lähettääkseen tietoja eteenpäin. Kuvassa 5 esitellään tietojen välittyminen palvelusta (*Virtual Server*) halutulle tietojen vastaanottajalle (*Pool of IPFIX Collectors*). Liitteessä 1 kuvataan säännöstö, jolla tutkitaan laitteen läpi kulkevaa jonkin tietyn palvelun HTTP-liikennettä ja poimitaan kunkin yhteyden aloitusaika, lähdeosoite, lähdeportti ja yhteyden kesto. Liitteen 1 rivillä 16 määritellään kohdassa 2.1 esitelty malline ja mallineessa käytettävät informaatioelementit. Riveillä 25–27 informaatioelementteihin lisätään kuormanjakajan funktioiden avulla liikenteestä halutut tiedot.

IANA:n määrittelemien informaatioelementtien lisäksi tuotteesta löytyy joitakin F5 Networksin määrittelemiä yrityskohtaisia elementtejä. On myös mahdollista lisätä itse omia yrityskohtaisia informaatioelementtejä. Lisääminen tapahtuu listauksen 1 kuvaamalla tavalla. Esimerkissä lisätään HTTP-pyynnön polku, metodi ja asiakasohjelma. Informaatioelementtejä voidaan tämän jälkeen käyttää liitteen 1 mukaisissa säännöstöissä, jos haluttu tieto saadaan poimittua liikenteestä jollain funktiolla.



Kuva 5. BIG-IP Local Traffic Manager ja tietojen välittyminen [F5 Networks 2016].

```
create sys ipfix element httpPath id 2 data-type string size 128 enterprise-id 65
create sys ipfix element httpMethod id 3 data-type string size 128 enterprise-id 65
create sys ipfix element httpUserAgent id 4 data-type string enterprise-id 65
```

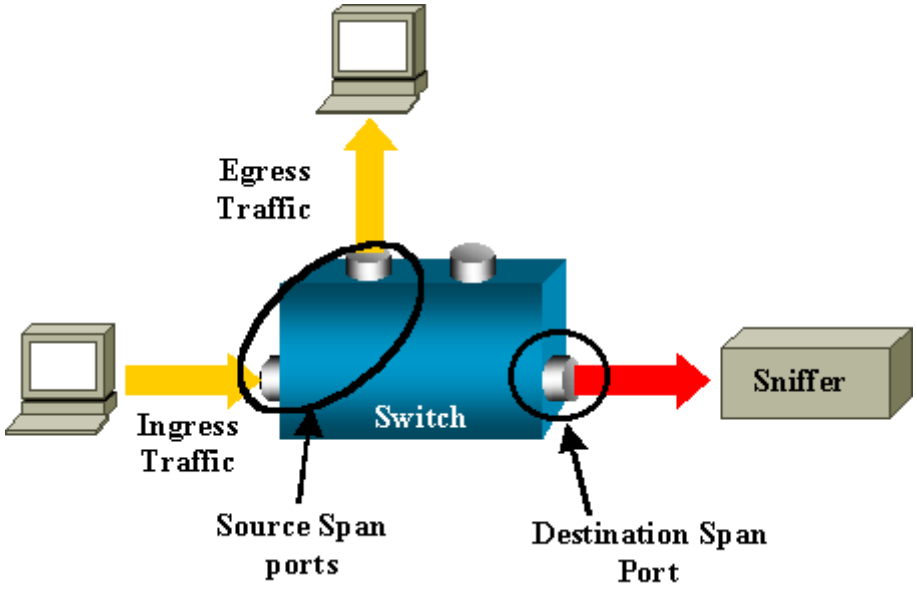
Listaus 1. BIG-IP Local Traffic Manager ja yrityskohtaisten informaatioelementtien lisääminen [F5 Networks 2016].

Käsitellään vielä toinen kuormanjakaja eli Citrix NetScaler. NetScaler tuotteessa IPFIX-protokollaa kutsutaan AppFlow-nimellä. Tuotteessa asetetaan päälle AppFlow-ominaisuus, joka kerää laitteen läpi kulkevasta liikenteestä halutut tiedot ja lähettää ne eteenpäin. Tuotteessa on IANA:n määrittelemien informaatioelementtien lisäksi joitakin Citrixin määrittelemiä yrityskohtaisia informaatioelementtejä. Tuote ei tue yrityskohtaisten informaatioelementtien lisäämistä itse [Citrix 2016].

Käsitellään viimeiseksi erilaiset tietoliikenneverkkoon sijoitettavat anturit. Antureille kopioidaan tietoverkkoliikenne esimerkiksi kytkimen portin peilauksen (*port mirroring*) avulla, jonka toimintaperiaate esitellään kuvassa 6. Antureiden keräämää tietoa voidaan

analysoida antureiden omilla työkaluilla, jotka ovat antureilla itsellään tai jollakin ulkoisella palvelimella, tai tiedot voidaan lähettää erilliselle keräimelle käyttäen esimerkiksi IPFIX-protokollaa. Markkinoilla olevia antureita ovat esimerkiksi Flowmon Probe, ntop nBox ja Telesoft Technologies IPFIX Probe. Flowmon-yrityksellä on Probe-tuotteen ohella erillinen Collector-niminen tuote, joka toimii keräimenä tiedoille [Flowmon 2016].

Anturilta voidaan lähettää tiedot myös jollekin toiselle keräimelle. Ntop-yrityksen nBox-tuote käyttää myöhemmin mainittavaa ja vertailtavaa nProbe-ohjelmistoa. nBox-tuotetta voidaan käyttää samanaikaisesti anturina ja keräimenä tai sieltä voidaan lähettää tiedot jollekin toiselle keräimelle [ntop 2016]. Telesoft Technologies -yrityksen IPFIX Probe -tuote toimii vain anturina, josta lähetetään tiedot eteenpäin keräimille [Telesoft Technologies 2016].



Kuva 6. Kytkimen portin peilauksen toimintaperiaate [Cisco 2014].

3.2. Tietojen vastaanottaja

Analysoitavan tiedon vastaanottajana eli keräimenä toimii yksittäinen sovellus tai ohjelmisto, josta tiedot välitetään toiselle sovellukselle tai ohjelmistolle analysoitavaksi. Ohjelmistot toimivat usein siten, että niissä on omat prosessit tietojen vastaanottoon, rikastukseen ja visualisointiin. Eri ohjelmat tai ohjelmistot voivat toimia tuotteesta tai järjestelmän suunnittelusta riippuen hajautetusti eri palvelimilla tai kaikki samalla palvelimella. Monesti tuotteisiin saa ladattua erillisiä liitännäisiä tuomaan lisäominaisuuksia. Markkinoilla on olemassa useita kaupallisia ohjelmistoja, joita kutsutaan *Security Information and Event Management* (SIEM) -tuotteiksi.

Käsitellään lyhyesti SIEM-tuotteiden tuki IPFIX-protokollalle ja yrityskohtaisille informaatioelementeille. Esimerkkeinä kaupallisista tuotteista ovat tutkimusyhtiö Gartnerin SIEM-nelikentän johtavat tuotteet: HP ArcSight, IBM QRadar, LogRhythm, McAfee (Intel Security) Enterprise Security Manager ja Splunk [Kavanagh and Rochford 2015]. Kaupallisten tuotteiden lisäksi markkinoilla on joitakin avoimen lähdekoodin toteutuksia SIEM-järjestelmistä. Esimerkkeinä näistä ovat AlienVault OSSIM ja LOGalyze. OSSIM eli *Open Source SIEM* on AlienVaultin kaupallisesta *Unified Security Management* (USM) -tuotteesta tehty avoimen lähdekoodin versio. OSSIM koostuu useista aktivoitavista sovelluksista, joita ovat esimerkiksi fprobe, nfdump ja ntop [Alienvault 2016].

Taulukossa 1 kuvataan SIEM-tuotteet ja vertaillaan niiden tukea IPFIX-protokollalle sekä yrityskohtaisille informaatioelementeille. Lähes kaikki Gartnerin SIEM-nelikentän johtavista tuotteista tukevat IPFIX-protokollaa, mutta vain osassa on tuki myös yrityskohtaisille informaatioelementeille. Vertailuun valituista avoimen lähdekoodin tuotteista vain OSSIM tukee myös IPFIX-protokollan yrityskohtaisia informaatioelementtejä.

Tuote	Kaupallinen	IPFIX-tuki	*Enterprise-specific* -kenttien tuki
AlienVault OSSIM	Ei	Kyllä	Kyllä
HP ArcSight	Kyllä	Ei tiedossa	Ei tiedossa
IBM QRadar	Kyllä	Kyllä	Kyllä
LOGalyze	Ei	Ei	Ei
LogRhythm	Kyllä	Ei tiedossa	Ei tiedossa
McAfee ESM	Kyllä	Kyllä	Ei tiedossa
Splunk	Kyllä	Kyllä	Kyllä

Taulukko 1. SIEM-tuotteiden tuki IPFIX-protokollalle ja *enterprise-specific* -kentille.

SIEM-järjestelmien ohella on myös useita tietoja vastaanottavia sovelluksia ja ohjelmistoja. Osassa on myös mahdollisuus tietojen visualisointiin. Velan [2013] tutki joitakin avoimen lähdekoodin ohjelmistoja ja niiden tukea IPFIX-protokollalle sekä yrityskohtaisille informaatioelementeille. Tutkimuksen aikaan ohjelmistoista moni tuki IPFIX-protokollaa, mutta tukea yrityskohtaisille informaatioelementeille ei ollut. Tällä hetkellä tilanne on hieman parantunut sovelluksien kehityksen ja IPFIX-protokollan yleistymisen ansiosta. Lisäksi uusia sovelluksia on kehitetty.

Taulukossa 2 kuvataan erillisiä sovelluksia ja vertaillaan niiden tukea IPFIX-protokollalle sekä yrityskohtaisille informaatioelementeille. Kaikki valitut tuotteet paitsi fprobe tukevat IPFIX-protokollaa ja moni tukee myös yrityskohtaisia informaatioelementtejä. SiLK-ohjelmiston osittainen tuki yrityskohtaisille informaatioelementeille tarkoittaa, että tekijät

ovat lisänneet omia informaatioelementtejä, mutta käyttäjät eivät voi lisätä niitä muuttamatta ohjelmakoodia.

Tuote	IPFIX-tuki	*Enterprise-specific* -kenttien tuki
fprobe	Ei	Ei
IPFIXcol	Kyllä	Kyllä
logstash	Kyllä	Kyllä
nfdump	Kyllä	Ei
nProbe/nTop	Kyllä	Kyllä
SiLK	Kyllä	Osittainen
Vermont	Kyllä	Ei

Taulukko 2. Avoimen lähdekoodin sovellusten ja ohjelmistojen tuki IPFIX-protokollalle ja *enterprise-specific* -kentille.

4. Sovelluskerroksen tiedot

4.1. HTTP- ja HTTPS-liikenteen analysointi ja hyökkäykset

Käytetään esimerkkeinä sovelluskerroksen protokollia *Hypertext Transfer Protocol* (HTTP) ja *Hypertext Transfer Protocol over TLS* (HTTPS). Protokollien erona on, että HTTPS on salattu käyttäen TLS- tai SSL-protokollia ja HTTP on salaamaton. Tämän vuoksi HTTPS-liikennettä voidaan analysoida vain sellaisten laitteiden avulla, jotka kykenevät purkamaan salauksen tai joihin salattu yhteys päätetään. Vuorovaikutusmallina protokollissa on erilliset pyynnöt ja vastaukset. Asiakasohjelma lähettää isäntäpalvelimelle pyynnön, jonka jälkeen isäntäpalvelin lähettää vastauksen asiakasohjelmalle.

Protokollia käytetään erityisesti WWW-sivuja selatessa. Protokollien liikenteen tutkiminen on mielenkiintoista ja tärkeää juuri niiden yleisyyden takia. Tietoja analysoimalla voidaan esimerkiksi havaita WWW-sivustoihin kohdistuvia hyökkäyksiä, selvittää WWW-sivustojen mahdollisia suorituskykyyn liittyviä ongelmia, tunnistaa ei-toivottuja virhetilanteita tai luoda tilastoja eri kävijöistä.

Husák ja muut [2015] ja van der Toorn ja muut [2015] ovat tutkineet, kuinka erilaisia WWW-sivustoihin kohdistuvia hyökkäyksiä voidaan tunnistaa ja valvoa sovelluskerroksesta poimittavien tietojen avulla hyödyntäen IPFIX-protokollaa. Molemmissa tutkimuksissa yhtenä hyökkäystyyppinä oli erilaiset niin kutsutut sanakirjahyökkäykset (*dictionary attack)* ja raa'an voiman hyökkäykset (*brute-force attack*).

Sanakirjahyökkäykset perustuvat valmiiksi laadittujen sanojen tai sanayhdistelmien ja raa'an voiman hyökkäykset satunnaisesti eri merkkien syöttämiseen lomakkeeseen tai kirjautumissivuun. Menetelmillä pyritään arvaamaan tietyn käyttäjän salasana. Mahdollisia kirjautumismekanismeja ovat esimerkiksi HTTP:n *Basic Authentication*, lomakepohjainen

kirjautuminen ja erilaiset rajapinnat kuten XML-RPC. Taulukossa 3 kuvataan kolmen eri julkaisujärjestelmän sisäänkirjautumissivun osoite ja tarvittavat tiedot kirjautumiseen.

CMS	URL	Attributes
Wordpress	/wp-login.php	username, password
	/xmlrpc.php	username, password
Joomla	/administrator/index.php ⊢ ?option=com_login	username, password access token, task
Drupal	/?q=user /?q=user/login	username, password access token, task
	/xmlrpc.php	username, password

Taulukko 3. Julkaisujärjestelmien kirjautumismekanismit [van der Toorn *et al.* 2015].

Hyökkäyksen tunnistus voidaan tehdä esimerkiksi seuraavaksi kuvatulla tavalla. Kerätään HTTP- tai HTTPS-pyyntöjen otsikoista julkaisujärjestelmän sisäänkirjautumissivu ja pyynnön vastauskoodi. Tämän jälkeen kootaan yhteen kaikki tietystä lähdeosoitteesta lähetetyt pyynnöt. Jos havaitaan, että samasta lähdeosoitteesta on lähetetty useita pyyntöjä ja kaikkiin tai lähes kaikkiin on saatu vastauskoodina *401 Unauthorized*, voidaan todeta kyseessä olevan mitä luultavimmin hyökkäys [van der Toorn *et al.* 2015]. Analysointia voidaan vielä laajentaa keräämällä lähetetty käyttäjätunnus, salasana ja mahdolliset muut muuttujat joko pyynnön otsikoista tai hyötykuormasta.

Husák ja muut [2015] lähestyivät sanakirja- ja raa'an voiman hyökkäyksien tunnistamista tutkimalla muuttujien arvojen sijaan muuttujien kardinaliteettia keskenään. Liikenteen he luokittelivat siten, että muuttujina olivat lähdeosoite, kohdeosoite ja HTTP-pyyntö. Liikenteen luokkajako muodostui kolmeen eri luokkaan. Luokka I kuvaa liikennettä, jossa yhdestä lähteestä lähetetään yhteen kohteeseen yhdenlaisia pyyntöjä. Luokka II kuvaa liikennettä, jossa yhdestä lähteestä lähetetään useaan eri kohteeseen yhdenlaisia pyyntöjä. Luokka III kuvaa liikennettä, josta yhdestä lähteestä lähetetään useaan eri kohteeseen useanlaisia pyyntöjä. Aikaisemmin mainitut hyökkäystyypit voitiin havaita luokan I mukaisena liikenteenä.

Taulukossa 4 kuvataan yhden päivän ajalta HTTP-liikenne koottuna lähdeosoitteen (*Guest*) mukaan. Suurin osa toistuneiden pyyntöjen HTTP-polusta sisälsi merkkijonon *admin* tai *login*, mikä viittaa juurikin edellä mainittuihin hyökkäyksiin. Aikaisemmin käsiteltyjen hyökkäystyyppien lisäksi Husák ja muut [2015] tutkivat, miten voidaan tunnistaa erilaiset skannerit ja hakurobotit, joita käsitellään seuraavaksi.

TOP REPEATED REQUESTS FROM ONE DAY (DATA SET SUMMER 2014)

Guest	Host	HTTP Path	#Flows
G1	H1	/wp-**login**.php	46,031
G2	H2	/**admin**istrator/index.php	27,965
G3	H2	/**admin**istrator/index.php	27,798
G4	H3	/wp-**login**.php	25,316
G5	H4	/pub/linux/slax/Slax-7.x/7.0.8/slax-Chinese-Simplified-7.0.8-i486.iso	5,921
G6	H5	/**proxy**/lib**proxy**.pac	5,036
G7	H6	/node/	4,286
G8	H4	/pub/linux/slax/Slax-7.x/7.0.8/slax-English-US-7.0.8-i486.zip	4,170
G9	H7	/wp-**login**.php	3,632
G10	H7	/polit/wp-**login**.php	3,632

Taulukko 4. Suosituimmat pyynnöt päivän ajalta. [Husák *et al.* 2015].

4.2. HTTP- ja HTTPS-liikenteen analysointi ja skannereiden sekä hakurobottien tunnistaminen

Edellisessä luvussa esiteltyjen hyökkäysten tunnistuksen lisäksi voidaan tunnistaa erilaisia skannereita ja hakurobotteja analysoimalla HTTP- ja HTTPS-liikennettä. Skannereilla tarkoitetaan tässä yhteydessä ohjelmistoja, jotka etsivät useilta eri palvelimilta mahdollisesti haavoittuvia sovelluksia [Husák *et al.* 2015]. Tällaisia voivat olla edellisessä luvussa mainitut julkaisujärjestelmät, kuten WordPress, erilaiset keskustelufoorumit, kuten phpBB, tai WWW-sivujen ylläpitoon tarkoitetut hallintapaneelit, kuten cPanel.

Skannereiden liikenne on edellisessä luvussa kuvatun luokan II mukaista eli yhdestä lähteestä lähetetään useaan kohteeseen yhdenlaisia pyyntöjä. Oleellisin tieto pyynnöissä on HTTP-polku, joka esimerkiksi WordPress-julkaisualustan tapauksessa on taulukossa 4 kuvattu */wp-login.php*. Kun kootaan yhteen tietystä lähdeosoitteesta useaan kohteeseen lähetetyt tietyt pyynnöt, voidaan todeta kyseessä olevan mitä luultavimmin skanneri [Husák *et al.* 2015].

Tämä lähestymistapa aiheuttaa kuitenkin ongelmia kuten Husák ja muut [2015] huomasivat. Tavallisen käyttäjän HTTP- tai HTTPS-liikenne voi näyttää skannerilta, koska useimpien WWW-sivujen latauksessa ladataan myös esimerkiksi WWW-sivulle määritelty kuvake eli *favicon*. Tämän syystä analysointivaiheessa on tärkeää olla jonkinlainen listaus tai muu tietämys eri sovelluksista, jotta tuloksista voidaan suodattaa pois säännönmukainen liikenne. Käsitellään seuraavaksi luvun alussa mainitut hakurobotit.

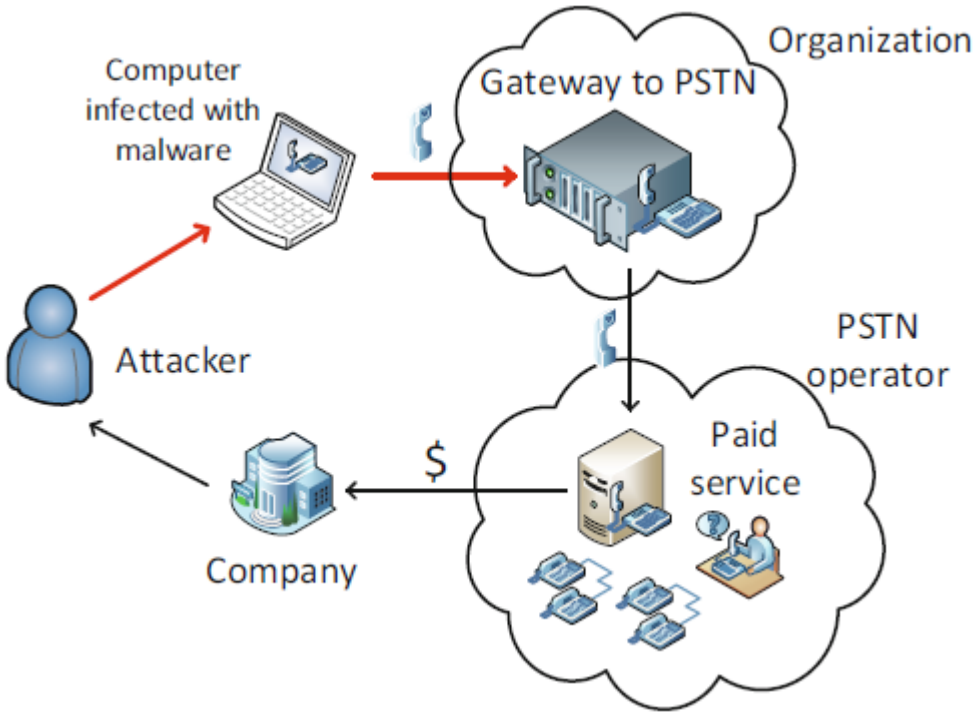
Hakurobotit pyytävät systemaattisesti WWW-sivustojen alisivuja sekä seuraavat syvemmälle niiden linkkejä. Hakuroboteilla tarkoitetaan joko hakukoneiden robotteja tai yksittäisiä ohjelmistoja, joita voidaan käyttää esimerkiksi hyökkäysvektoreiden etsintään jostakin tietystä WWW-sivustosta. Hakurobottien liikenne ilmenee edellisessä luvussa kuvatun luokan III mukaisena eli yhdestä lähteestä lähetetään useaan eri kohteeseen useanlaisia pyyntöjä. Luokan III mukaisen liikenteen haasteena kuitenkin on, että tavallisten käyttäjien liikenne muistuttaa juuri tällaista liikennettä.

Tämän vuoksi analysoidessa tietoja tulee tutkia esimerkiksi lähdeosoitteen käänteisnimipalvelunimi tai HTTP-pyyntöjen asiakasohjelma, jotta hakurobotit voidaan tunnistaa [Husák *et al.* 2015]. Asiakasohjelma kertoo tunnettujen hakurobottien tapauksessa kyseisen yrityksen nimen. Tosin on huomattava, että HTTP-pyyntöjen otsikoita voi muuttaa ja siten HTTP-pyynnössä lähetettyä asiakasohjelmaa ei voida pitää täysin luotettavana tietona.

4.3. Muiden sovelluskerroksen protokollien liikenteen analysointi

Käsitellään vielä HTTP- ja HTTPS-liikenteen lisäksi *Voice over IP* (VoIP) -liikennettä eli niin kutsuttuja Internet-puheluita. Puheluiden aloitukseen, hallintaan ja lopetukseen käytetään esimerkiksi sovelluskerroksen protokollaa *Session Initiation Protocol* (SIP). Protokolla on vuorovaikutusmalliltaan samankaltainen kuin kohdassa 4.1. kuvatut HTTP ja HTTPS. Erona on, että laitteet voivat toimia sekä asiakasohjelmina että isäntäpalvelimina. Verkossa erilliset välityspalvelimet (*proxy server*) toimivat puheluiden yhdistäjinä ja yrityksien sisällä välityspalvelimia kutsutaan nimellä *Private Branch Exchange* (PBX). Näiden kautta kulkee usein myös tavallinen puhelinyhteys *Public Switched Telephone Network* (PSTN). [Cejka *et al.* 2015]

Cejka ja muut [2015] tutkivat väärinkäyttöä, jossa heikosti konfiguroitu PBX sallii Internet-puheluiden soittamisen PSTN-numeroihin. Tällöin hyökkääjä pystyy soittamaan erilaisiin maksullisiin numeroihin ansaitakseen rahaa ja tuottamaan hyväksikäytettävän PBX:n omistajalle taloudellisia menetyksiä. Tämän tyyppinen hyökkäys esitetään kuvassa 7.



Kuva 7. SIP-petoksen periaatekuva [Cejka *et al.* 2015].

Väärinkäyttöjen ja hyökkäysten tunnistus tapahtuu analysoimalla SIP INVITE -pyyntöjä, joilla yritetään soittaa PSTN-numeroihin. Koska soittamiseen PSTN-numeroon vaaditaan erillinen etuliite, jota hyökkääjän on mahdotonta selvittää, tarvitsee se yrittää arvata esimerkiksi raa'an voiman menetelmällä. Tunnistaminen voidaan tehdä kokoamalla yhdestä lähdeosoitteesta tiettyyn kohdeosoitteeseen tulevat pyynnöt, joissa kohteen etuliite vaihtelee [Cejka *et al.* 2015]. Menetelmällä voidaan myös tunnistaa onnistuneet yritykset.

5. Yhteenveto

Sovelluskerroksen tietojen tutkiminen tietovirtojen avulla on vielä suhteellisen uusi lähestymistapa tietoverkkoliikenteen valvontaan ja analysointiin. Tämän vuoksi monet laitteet ja ohjelmistot eivät vielä täysin tue IPFIX-protokollaa ja kaikkia sen ominaisuuksia. Suurimpia haasteita IPFIX-protokollan käytössä on juurikin laitteiden ja ohjelmistojen kyvykkyys tai sen puute poimia tietoverkkoliikenteestä halutut tiedot. Jatkuva tutkimustyö ja avoimen lähdekoodin ratkaisut varmasti vauhdittavat monien verkkolaitteiden ja ohjelmistojen valmistajia kehittämään tuotteidensa tukea IPFIX-protokollalle ja sen suomille mahdollisuuksille.

Jatkossa ja osittain tälläkin hetkellä vielä ongelmallisempaa on kuitenkin joidenkin protokollien vuorovaikutusmalli ja liikenneprofiili. Ongelmallisina tapauksina ovat esimerkiksi viime vuosina paljon suosiota saaneet musiikin ja videoiden suoratoistopalvelut, joissa asiakasohjelman ja isäntäpalvelimen välillä vaihdetaan suuri määrä paketteja useiden minuuttien ja jopa tuntien aikana. Tutkielmassa esitellyt protokollat HTTP, HTTPS ja SIP perustuvat vuorovaikutukseltaan yksittäisiin pyyntöihin ja niihin liittyviin vastauksiin.

Yleisesti ottaen IPFIX-protokollan käyttäminen sovelluskerroksen tietojen tutkimiseen on käyttökelpoinen ja hyvä tapa. Tätä väittämää tukee monet aiheesta tehdyt tutkimukset ja prototyypit, vaikka itse protokolla onkin kohtalaisen uusi.

Viiteluettelo

Alienvault. 2016. *OSSIM: The Open Source SIEM*. Available at https://www.alienvault.com/products/ossim. Checked 9.4.2016.

Tomas Cejka, Vaclav Bartos, Lukas Truxa and Hana Kubatova. 2015. Using Application-Aware Flow Monitoring for SIP Fraud Detection. In: Steven Latré, Marinos Charalambides, Jérôme François, Corinna Schmitt and Burkhard Stiller (eds.), *Intelligent Mechanisms for Network Configuration and Security*. Springer, 87-99.

Cisco. 2014. *Catalyst Switched Port Analyzer (SPAN) Configuration Example*. Available: http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/10570-41.html. Checked: 24.5.2016.

Citrix. 2016. *Configuring the AppFlow Feature*. Available at http://docs.citrix.com/en-us/netscaler/11/system/ns-ag-appflow-intro-wrapper-con.html. Checked 7.5.2016.

F5 Networks. 2016. *Customizing IPFIX Logging with iRules*. Available at https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/bigip-external-monitoring-implementations-12-0-0/12.html. Checked 7.5.2016.

Flowmon. 2016. *Flowmon - Probe*. Available: https://www.flowmon.com/en/products/flowmon/probe. Checked 6.3.2016.

Martin Husák, Petr Velan and Jan Vykopal. 2015. Security monitoring of HTTP traffic using extended flows. In: *Proc. of the 10th International Conference on Availability, Reliability and Security (ARES)*, 258-265.

IANA. 2016a. *Private Enterprise Numbers*. Available at http://www.iana.org/assignments/ enterprise-numbers/enterprise-numbers. Checked 5.3.2016.

IANA. 2016b. *IP Flow Information Export (IPFIX) Entities*. Available at http://www.iana.org/assignments/ipfix/ipfix.xhtml. Checked 5.3.2016.

Kelly M. Kavanagh and Oliver Rochford. 2015. *Magic Quadrant for Security Information and Event Management*. Gartner.

ntop. 2016. *nProbe – An Extensible NetFlow v5/v9/IPFIX Probe for IPv4/v6*. Available: http://www.ntop.org/products/netflow/nprobe. Checked 9.4.2016.

RFC 7011. 2013. *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information*. Available at http://tools.ietf.org/html/rfc7011. Checked 5.3.2016.

RFC 7012. 2013. *Information Model for IP Flow Information Export (IPFIX)*. Available at http://tools.ietf.org/html/rfc7012. Checked 5.3.2016.

Telesoft Technologies. 2016. *IPFIX Probe - Cyber Defence and Network Visibility*. Available at http://telesoft-technologies.com/technologies/cyber-security/ipfix-probe. Checked 9.5.2016.

Olivier van der Toorn, Rick Hofstede, Mattijs Jonker and Anna Sperotto. 2015. A first look at HTTP(S) intrusion detection using netflow/IPFIX. In: *Proc. of the 2015 IFIP/IEEE International Symposium on Integrated Network Management*, 862-865.

Petr Velan. 2013. Practical experience with IPFIX flow collectors. In: *2013 IFIP/IEEE International Symposium on Integrated Network Management*, 1021-1026.

Petr Velan and Pavel Čeleda. 2014. Next generation application-aware flow monitoring. In: *Proc. of the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS)*, 173-178.

Wireshark. 2016. *Wireshark Display Filter Reference: Cisco NetFlow/IPFIX*. Available at https://www.wireshark.org/docs/dfref/c/cflow.html. Checked 24.5.2016.

Tietyn palvelun HTTP-liikenteen tutkiva säännöstö (iRule) F5 BIG-IP Local Traffic Manager -laitteessa

```
1 # This rule captures HTTP traffic and sends logs to IPFIX collectors.
2
3  when RULE_INIT {
4    set static::http_rule1_dest ""
5    set static::http_rule1_tmplt ""
6  }
7
8  when CLIENT_ACCEPTED {
9    if { $static::http_rule1_dest == ""} {
10     # open the logging destination if it has not been opened yet
11     set static::http_rule1_dest [IPFIX::destination open -publisher /Common/ipfix_publisher]
12   }
13
14   if { $static::http_rule1_tmplt == ""} {
15     # if the template has not been created yet, create the template
16     set static::http_rule1_tmplt [IPFIX::template create "flowStartSeconds sourceIPv4Address tcpSourcePort flowDurationMilliseconds"]
17    }
18 }
19
20  when HTTP_REQUEST {
21   # create a new message for this request
22   set rule1_msg1 [IPFIX::msg create $static::http_rule1_tmplt]
23
24   # compose the IPFIX log message
25   IPFIX::msg set $rule1_msg1 flowStartSeconds [clock seconds]
26   IPFIX::msg set $rule1_msg1 sourceIPv4Address [IP::client_addr]
27   IPFIX::msg set $rule1_msg1 tcpSourcePort [TCP::client_port]
28
29   # record the start time in milliseconds
30   set start [clock clicks -milliseconds]
31  }
32
33  when HTTP_RESPONSE_RELEASE {
34   # figure out the final duration and add it to the IPFIX log
35   set stop [expr {[clock click -milliseconds] - $start}]
36   IPFIX::msg set $rule1_msg1 flowDurationMilliseconds $stop
37
38   # send the IPFIX log
39   IPFIX::destination send $static::http_rule1_dest $rule1_msg1
40  }
41 }
```

Tietyn palvelun HTTP-liikenteen tutkiva säännöstö (iRule) F5 BIG-IP Local Traffic Manager -laitteessa [F5 Networks, 2016].

Eettinen näkökulma autonomisiin tieliikenneajoneuvoihin

Sami Voutilainen

Tiivistelmä.
Itseohjautuvat autot tekevät tuloaan tieliikenteeseen, mutta monet niiden toimintaa koskevat eettiset kysymykset ovat yhä avoinna. Tässä tutkielmassa pureudutaan näihin kysymyksiin tutkimalla sekä järjestelmän autonomisuuden, eli itsenäisen toimintakyvyn, että etiikan ohjelmoinnin vaikeuksia. Lisäksi selvitetään miten vastuu voisi jakautua kuljettajan sekä itseohjautuvan ajoneuvon valmistajan välillä.

Tutkielmaa varten analysoitiin useita ajoneuvojen autonomisoinnin kannalta relevantteja teoksia, joiden pohjalta pystyttiin luomaan käsitys itseohjautuvien ajoneuvojen valmiuksien puutteellisuudesta. Autonomisuuden kannalta teknologia on kehittynyt huomattavasti, ja nykyisin itseohjautuvat ajoneuvot pystyvätkin toimimaan jo hyvin itsenäisesti. Autonomisuuteen kuuluu kuitenkin teknisten toteutuksien lisäksi myös eettinen rajapinta, sillä ajoneuvot voivat tieliikenteessä joutua tekemään vaikeita moraalia vaativia päätöksiä. Etiikan ohjelmoinnissa on vielä useita kompastuskiviä, joita on ratkaistava ennen itseohjautuvuuden saapumista tieliikenteeseen. Lisäksi autonomisia ajoneuvoja koskeva lainsäädäntö laahaa teknologisien mahdollisuuksien perässä. Itseohjautuvia ajoneuvoja ajatellessa onkin muistettava, että aihe on teknisyydessään pullollaan myös eettisiä sekä lainsäädännöllisiä ongelmia.

Avainsanat ja -sanonnat: itseohjautuva auto, automaatio, etiikka, moraali

1. Johdanto

Tässä tutkielmassa keskitytään itseohjautuvien ajoneuvojen eettisiin ongelmakohtiin. Aihe on tärkeä, sillä useat ajoneuvovalmistajat suunnittelevat jo omia itseohjautuvia autojaan. Ajoneuvojen suunnittelussa tarvitaan ymmärrystä eettisistä periaatteista, sillä itseohjautuvan auton on tärkeää olla kyvykäs toimimaan esimerkiksi vaikeissa tieliikenneonnettomuustilanteissa. Tietojenkäsittelytieteellisesti etiikan teorioiden soveltaminen on kuitenkin vielä lapsenkengissään. Tämän lisäksi itseohjautuvan ajoneuvon käsite on lainsäädännölle tuntematon, vaikka tiedettävästi ensimmäinen ajoneuvon aiheuttama onnettomuus tapahtui vastikään helmikuussa 2016 [Google 2016]. Tutkielmassa pyritään kartoittamaan näitä eettisiä ja lainsäädännöllisiä ongelmia sekä esittämään niihin ehdotettuja ratkaisuvaihtoehtoja.

Tutkielma aloitetaan selittämällä muutama sisällön kannalta oleellinen termi. Sen jälkeen puhutaan itseohjautuvien autojen tämänhetkisestä tilanteesta ja käytännön toteutuksista, etenkin Googlen implementaatiota tarkastellen. Nykytilanteesta siirrytään tarkastelemaan tulevaisuuden visioita niin kone-etiikan kuin itseohjautuvien autojenkin alalla, josta edetään pohtimaan, kenellä on vastuu itseohjautuvan auton aiheuttaessa onnettomuuden.

2. Autonomiset agentit

Tämän tutkielman keskipisteessä ovat niin sanotut autonomiset agentit. Autonomisuus voitaisiin lyhyesti määritellä järjestelmän kyvyksi toimia itsenäisesti. Agentti puolestaan on autonomisesti toimiva entiteetti. Seuraavat kohdat pyrkivät selittämään autonomisuuden ja agentin käsitteen perusteellisemmin sekä yleisesti että tässä tutkielmassa erityisesti ajoneuvoja koskevassa merkityksessä.

2.1. Autonomisuus

Sanaa autonomia käytetään tavallisesti valtiotieteissä ja sivistyssanakirja [2015] määrittelee sen sisäiseksi itsehallinnoksi tai valtiolliseksi itsemääräämisoikeudeksi, joka on täyttä itsenäisyyttä suppeampi. Sanan perinteisintä merkitystä mukaillen myös tietojenkäsittelytieteellisen järjestelmän autonomisuus tarkoittaa sen kykenevyyttä toimia itsenäisesti sille asetettujen sääntöjen puitteissa.

Autonominen järjestelmä voi olla täysin itsenäinen, mutta usein sen toimintaa valvoo määrätty korkeampi auktoriteetti. Useiden aihetta koskevien teorioiden mukaan ihmisellä tulisi aina olla viimeinen sana turvallisuutta koskevissa päätöksissä [Richards and Stedmon 2015]. Ajoneuvojen kannalta tämä tarkoittaa, että auton etenemistä valvova taho voi ohittaa järjestelmän generoimat päätökset esimerkiksi ottamalla kulkuneuvon manuaaliseen ohjaukseensa.

Autonominen järjestelmä ei ole sama asia kuin automaattinen järjestelmä, vaikka termejä saatetaan joissain yhteyksissä käyttää ristiin. Automaattinen järjestelmä suorittaa tarkoin määriteltyjä tehtäviä tunnetussa ympäristössä, kun taas autonominen järjestelmä pyrkii keräämään dataa ja mukautumaan toimimaan sille ennalta tuntemattomassa ympäristössä [Boissier et al. 2015]. Tärkeä autonomisen järjestelmän piirre on myös sen kyky suorittaa toimintoja ilman ihmiskäyttäjän välitöntä läsnäoloa [Richards and Stedmon 2015]. Toisaalta autonominen järjestelmä toimii hyvin harvoin vailla minkäänlaista ihmiskäyttäjän valvontaa [Boissier et al. 2015].

2.2. Agentit

Autonomisesti toimivia yksikköjä voidaan kollektiivisesti nimittää agenteiksi. Perinteisesti agentilla tarkoitetaan entiteettiä, joka kykenee toimimaan tai suo-

rittamaan sille annetun tehtävän [Boissier et al. 2015]. Ohjelmistotuotannon näkökulmasta Jennings [1999] luonnehtii agenttia kapseloiduksi tietokonejärjestelmäksi, joka sijaitsee sille asetetussa ympäristössä ja kykenee joustaviin, autonomisiin toimintoihin saavuttaakseen sille ohjelmoidun päämäärän. Franklinin ja Graesserin [1996] laajemman määritelmän mukaan agentti on järjestelmä, joka vaikuttaa ympäristössään ja toimii havaintojensa perusteella saavuttaakseen omat tavoitteensa ja jonka päätökset vaikuttavat myös tulevaisuuden havaintoihin. Franklinin ja Graesserin esittämään määritelmään sopivat myös ihmiset, minkä vuoksi on tärkeää tehdä ero ihmisagenttien ja niin sanottujen tekoälyllisten agenttien välille. Ihmisagentilla voidaan viitata käyttäjään eli henkilöön, joka toimii tekoälyllisen agentin kanssa tai auttaa sitä saavuttamaan tavoitteensa [Boissier et al. 2015]. Tekoälyllinen agentti taas viittaa ihmisen luomaan järjestelmään, joka täyttää aiemmin käsitellyt määritelmät. Ajoneuvoja ajatellen ihmisagentti olisi siis autonomisen ajoneuvon kuljettaja tai järjestelmänvalvoja ja ajoneuvo tekoälyllinen agentti. Tekoälyllistä ajoneuvojärjestelmää nimitetään tässä tutkielmassa myös ajoneuvoagentiksi.

Autonominen tieliikenneajoneuvo toimii aina ympäristössä, jossa vaikuttaa sen lisäksi muita agentteja, vähintään yksi valvova ihmisagentti. Ympäristöä, jossa sekä ihmisagentit että tekoälylliset agentit ovat vuorovaikuksessa, kutsutaan sosiotekniseksi järjestelmäksi [Boissier et al. 2015]. Ihmisagentin lisäksi tekoälylliset agentit voivat kommunikoida keskenään ja muodostaa näin niin sanottuja agenttien yhteisöjä [Franklin and Graesser 1996]. Jos tieliikenne kehitettäisiin täysin autonomiselle tasolle, voisivat ajoneuvoagentit luoda yhteisöjä, jossa ajoneuvot keskustelevat keskenään. Tällöin esimerkiksi suuntavilkun näyttäminen voitaisiin tehdä tarpeettomaksi, sillä ajoneuvot pystyisivät kommunikoimaan aikeistaan muin tavoin.

3. Ajoneuvo autonomisena agenttina

Teknologian kehityksen myötä ajatus autonomisuudesta on levinnyt myös autoteollisuuteen. Useat valmistajat ovat ottaneet jo käyttöön erilaisia ajamista helpottavia järjestelmiä [Richards and Stedmon 2015]. Tällaisia ovat esimerkiksi mukautuva vakionopeudensäädin, joka ylläpitää ajonopeutta säilyttäen sopivan etäisyyden edellä ajavaan ajoneuvoon, pysäköimistä helpottavat järjestelmät, omalla kaistalla pysymisessä auttava kaistavahti sekä törmäyksenvälttöjärjestelmät [Fagnant and Kockelman 2015]. Autonomisuudella pyritään parantamaan liikenneturvallisuutta ja vähentämään tieverkon ruuhkautumista sekä kasvihuonepäästöjä [Payre et al. 2014]. Täysin autonomisesti toimivaa autoa ei kuitenkaan markkinoilla vielä ole, vaikka useat tunnetut suuryritykset kuten Google, Toyota, Nissan ja BMW tekevät jatkuvaa tutkimusta aiheesta [Richards

and Stedmon 2015]. Autonomisuuden tutkimukseen liittyy useita käytettävyyteen sekä eettisyyteen liittyviä ongelmakohtia, joista etenkin eettistä ulottuvuutta pyritään käsittelemään tässä tutkielmassa. Ensin kuitenkin on oleellista luoda katsaus autonomisen ajoneuvon toimintavaatimuksiin sekä tämänhetkisiin toteutuksiin, minkä jälkeen pohditaan autonomisuuteen liittyviä ongelmia.

3.1. Käytäntö

Autonominen ajoneuvo vaatii toimiakseen useita sensoreita, joilla se saa tarvittavan tarkan käsityksen ympäristöstään. Esimerkiksi teknologiayhtiö Googlen valmistama itseohjautuva auto käyttää toimiessaan 360-asteista laserskanneria esteiden havaitsemiseen, neljää tutkaa muun liikenteen tarkkailuun, yhtä erillistä kameraa liikennevalojen seurantaan ja GPS-paikanninjärjestelmää ajoneuvon sijainnin tarkkaan määrittämiseen [Guizzo 2011]. Ajoneuvon käsittelyssä helpottavat autonomiset järjestelmät kuten mukautuva vakionopeudensäädin, törmäyksenvälttöjärjestelmä sekä pysäköintiavustaja toteutetaan usein ultraääniteknologialla [Richards 2015]. Hyvin varustellut autonomiset ajoneuvot voivatkin keräämänsä informaation perusteella pystyä tekemään mahdollisesti jopa ihmistä parempia ajopäätöksiä yllättävissä tilanteissa [Fagnant and Kockelman 2015].

Googlen itseohjautuvalla autolla on ajettu vuodesta 2009 helmikuuhun 2016 mennessä autonomisesti lähes 4 miljoonaa kilometriä, joista yli puolet auto on ajanut autonomisesti kuljettajan valvoessa. Projektin alusta lähtien itseohjautuvat autot ovat olleet osallisena pienissä tieliikenneonnettomuuksissa [Google 2015], mutta tiettävästi ensimmäinen niiden aiheuttama kolari tapahtui helmikuussa 2016. Googlen helmikuun raportin mukaan ajoneuvo ajoi kaistaa vaihtaessaan linja-auton kylkeen noin 3 kilometrin tuntivauhdilla linja-auton kulkiessa noin 25 kilometriä tunnissa. Ajoneuvoa valvova kuljettaja uskoi bussin antavan tilaa itseohjautuvalle autolle, mutta linja-auto ei ollut tilanteessa väistämisvelvollinen. Törmäyksestä ei aiheutunut henkilövahinkoja. [Google 2016.]

Googlen [2015] itseohjautuvan auton testiraportista selviää, kuinka usein ajoneuvo on täytynyt palauttaa manuaaliseen ohjaukseen. Uusimmat tiedot ovat marraskuulta 2015, jolloin kuljettaja otti ohjat omiin käsiinsä 16 kertaa kuukauden aikana. Syitä irrottautumiseen olivat ajoneuvon tekemät ei-toivotut ohjausliikkeet, hälytysajoneuvo liikenteessä, huonot sääolosuhteet, ohjelmisto- ja laitteisto-ongelmat sekä ”holtittomasti käyttäytyvä agentti”. Ongelmien kuvauksien perusteella ilman valvontaa liikkuvia itseohjautuvia ajoneuvoja tuskin nähdään tieliikenteessä vielä välittömässä lähitulevaisuudessa. On kuitenkin olemassa tapoja, joilla ajoneuvo voi toimia autonomisesti myös ihmisen valvonnan alaisena.

3.2. Autonomisuuden tasot

Järjestelmän autonomisuus voidaan jakaa tasoihin sen mukaan, kuinka paljon vaikutusvaltainen tekoälyllinen agentti on ihmisagenttiin verrattuna. Taulukossa 1 esitellään Richardsin ja Stedmonin [2015] tekemä vertailu kolmen autonomisuuden tasoa kuvaavan viitekehyksen välillä. Tasot on jaoteltu numeerisesti agenteilla olevien valtuuksien mukaan. Asteikkojen alimmassa päässä on taso, jossa kaikki toiminnot ovat täysin ihmisagentin vastuulla. Viitekehyksien toisessa ääripäässä tekoälyllinen agentti toimii ilman ihmisagentin osallistumista päätöksiin. Tällaista järjestelmää voidaan kutsua myös täysautonomiseksi järjestelmäksi. Lisäksi kuvatut viitekehykset sisältävät joustavan auktoriteetin tasoja, joissa agenttien auktoriteetin vahvuus vaihtelee. Edellisen määritelmän mukaan voidaan autonomisuus jakaa karkeasti kolmeen tasoon: ihmisauktoriteettiseen, tekoälyauktoriteettiseen sekä joustavan auktoriteetin tasoon. Yhtä oikeaa perustelua joko ihmisauktoriteettisen tai tekoälyauktoriteettisen järjestelmän käyttämiseen on vaikea löytää; molemmat tasot sisältävät toiselta uupuvia vahvuuksia sekä puutteita.

Ihmisauktoriteettinen eli manuaalinen järjestelmä antaa nimensä mukaisesti ihmisagentin hoitaa kaikki järjestelmän toiminnot. Manuaalinen järjestelmä ei sisällä minkäänlaista autonomista aputeknologiaa eikä tarjoa kuljettajalle lainkaan avustusta [Richards and Stedmon 2015]. Nykyajoneuvot edustavat kuitenkin enemmän joustavan auktoriteetin tasoa, jossa autonominen järjestelmä hoitaa kuljettajaa avustavia toimintoja hänen niin toivoessaan.

Joustavan auktoriteetin tasolla ihmisagentti ja tekoälyllinen agentti neuvottelevat yhdessä oikeasta päätöksestä toisen ollessa hallitseva auktoriteetti. Jos ihminen on hallitseva auktoriteetti, tekoäly voi tarjota hänelle ajoavustusta tai ehdottaa muita toimintoja, jotka ihmisen on hyväksyttävä. Samoin tekoälyn ollessa hallitseva auktoriteetti voi ihminen antaa järjestelmälle syötteitä, mutta tekoäly päättää onko niiden toteutus tarpeellista.

Tekoälyauktoriteettisella, eli täysautonomisella tasolla ajoneuvoagentti pystyy toimimaan itsenäisesti eikä vaadi ihmiseltä syötteitä. Viitekehyksestä riippuen ajoneuvo voi olla täysin huomioimatta ihmissyötettä, mutta toisissa ihmisellä on aina oikeus keskeyttää ajoneuvojärjestelmän autonominen toiminta.

Viitekehyksien autonomisuuden tasoluvut eroavat toisistaan, vaikka jokaisessa suurin luku ilmaisee täysautonomista järjestelmää ja pienin ihmisauktoriteettista tasoa. Taulukkoon on merkattu toisiaan lähellä olevat tasot harmaan eri sävyinä. Tummin harmaa on täysautonominen taso, sen jälkeen joustavan auktoriteetin taso ja viimeisenä manuaalinen tai hyvin matalan automaatiotason järjestelmä.

Autonomisuuden taso	PACT-viitekehys	NHTSA-viitekehys	Tason kuvaus
10	5	4	Täysautonominen; automaattinen; täysin itseohjautuva automaatio
9	5	4	Automatisoitu järjestelmä informoi ihmistä vain halutessaan; automaattinen; täysin itseohjautuva automaatio
8	4	3	Automaatiojärjestelmä informoi ihmistä vain pyydettäessä; joustava; rajoitettu itseohjautuva automaatio
7	4	3	Automatisoitu järjestelmä toimii autonomisesti ja informoi ihmisvalvojaa pyytämättä; joustava; rajoitettu itseohjautuva automaatio
6	3	2	Ihmisvalvojalla on tunnetun aikavälin veto-oikeus ennen automaattisen toiminnan suorittamista; joustava; yhteisen toiminnan automaatio
5	2	2	Automaatiojärjestelmä suorittaa ehdottamansa toiminnon ihmisvalvojan hyväksyessä; joustava; yhteisen toiminnan automaatio
4	2	1	Automaatiojärjestelmä antaa yksittäisiä toiminta-ehdotuksia; joustava; toimintokohtainen automaatio
3	2	1	Automaatiojärjestelmä antaa rajatun valikoiman toimintaehdotuksia; joustava; toimintokohtainen automaatio
2	1	1	Automaatiojärjestelmä antaa täydellisen valikoiman toimintaehdotuksia; joustava; toimintokohtainen automaatio
1	0	0	Tekoäly ei tarjoa avustusta, ihminen suorittaa kaikki toiminnot; manuaalinen; ei automaatiota

Taulukko 1: Richardsin ja Stedmonin [2015] lajitteluun perustuva kolmen viitekehyksen vertailu.

Nykyisin ajoneuvoteknologiassa ihminen on aina hallitseva auktoriteetti. Autonomisuuden tasosta huolimatta ihmisagentin on aina valvottava liikennettä sekä tekoälyllisen agentin toimintaa [Banks and Stanton 2015], sillä hän on lain silmissä lähes aina vastuussa ajoneuvostaan, vaikka tekoälyllinen agentti huolehtisi ajamisesta. Esimerkiksi Yhdysvalloissa autonomiset ajoneuvot ovat sallittuja Nevadassa, Floridassa ja Kaliforniassa, mutta yksikään osavaltio ei salli kuljettajattomia autonomisia ajoneuvoja [Gurney 2015].

Ihmisauktoriteettisen tai joustavan auktoriteetin tason käyttöä voidaan perustella sillä, että ihminen on nykylainsäädännön mukaan kuljettajana aina vastuussa ajoneuvonsa liikkeistä. Toisaalta voidaan ajatella, että tekoälyllinen agentti pystyy tekemään paremman päätöksen äkillisessä vaaratilanteessa laskelmoidun ja valppaan olemuksensa ansiosta. Tekoälyllinen agentti ei väsy tai menetä keskittymiskykyään kuten ihmisagentti, eikä se voi saattaa itseään ajokunnottomaksi [Gurney 2015]. Tilastokeskuksen [2014] mukaan vuoden 2014 aikana Suomessa 229 ihmistä menehtyi ja 519 loukkaantui vakavasti tieliikenneonnettomuuksissa. Henkilövahinkoihin johtaneista tieliikenneonnettomuuksista 460 oli rattijuopumusonnettomuuksia. Maailmanlaajuisesti jopa 90 prosenttia liikenneonnettomuuksista tapahtuu kuljettajan virheen seurauksena [Fagnant and Kockelman 2015].

Ajoneuvojen joustavan auktoriteetin autonomiset ominaisuudet eivät kuitenkaan ole ainoastaan parantaneet ihmisten liikenneturvallisuutta, vaan niiden käyttö tuo mukanaan uudenlaisia riskejä. On huomattu, että esimerkiksi edellä ajavan auton ajonopeuteen mukautuvan vakionopeudensäätimen käyttäminen voi hidastaa kuljettajan reaktioaikaa jopa yli sekunnilla [Merat and Jamson 2009]. Autonomisuuden lisääminen vähentää väistämättä käyttäjän suoritettavana olevia toimintoja, joka voi edelleen heikentää hänen havainnointikykyään liikenteessä. Ajoneuvon hallintaan liittyvien tehtävien sijaan ihmisen on havainnoitava sekä liikenteessä että ajoneuvojärjestelmässä tapahtuvia muutoksia [Banks and Stanton 2015]. Ihmisen roolin passivoituminen vaikeuttaa myös hänen valmiuttaan tarttua ohjaimiin, mikäli ajoneuvojärjestelmä ei jostain syystä kykene toimimaan autonomisesti [Payre et al. 2014]. Lisäksi Payre ja muut [2014] tuovat esille autonomisen ajoneuvon väärinkäytön riskit, sillä ihmisen luopuessa ajotehtävistä voi hänelle muodostua kuva, ettei hän ole enää vastuussa ajoneuvostaan.

Tekoälyllinen agentti pystyy käsittelemään sen ympärillä tapahtuvia asioita laskelmoivasti ja lyhyessä ajassa tavalla, johon ihminen ei ikinä kykene. Jos autonominen ajoneuvo joutuu yllättävään vaaratilanteeseen, esimerkiksi hirvikolariin, on tekoälyllisellä järjestelmällä lähes aina ihmistä nopeampi reaktionopeus [Fagnant and Kockelman 2015]. Tekoälyllisen agentin ongelmana voidaan kuitenkin pitää sen kykenemättömyyttä vaikeisiin eettisiin päätöksiin. Vaaratilanteissa sekunnin murto-osa voi olla kolarin lopputuloksen kannalta äärimmäisen merkityksellinen, jolloin on mahdotonta odottaa ihmissyötettä. Sen vuoksi on tärkeää pohtia, miten autonominen ajoneuvo pystyisi toimimaan eettisesti vaikeissa tilanteissa ilman ihmisopastusta.

4. Kone-etiikka

Tekoälyn kehittyessä yhä lähemmäksi täysautonomisuutta kysymys siitä, miten suunnitella ja toteuttaa moraalinen tekoälyllinen agentti, on tullut entistä tärkeämmäksi [Allen et al. 2000]. Kysymys on eettinen siinä missä tietojenkäsittelytieteellinenkin, sillä esimerkiksi ajoneuvoihin sovellettuna puhutaan päätöksistä, joissa kyse on elämästä ja kuolemasta [Allen et al. 2006]. Moraalisen rajapinnan lisääminen autonomisiin agentteihin pitää sisällään lukuisia haasteita, eikä ole vielä edes täysin selvää, onko eettisiä ohjesääntöjä ymmärtävän tekoälyllisen agentin luominen mahdollista. Näiden haasteiden tutkimusta kutsutaan yleisesti kone-etiikaksi.

Moraalinen päätöksenteko on monimutkainen prosessi, jota harva ihminenkään pystyy suorittamaan täysin virheettömästi [Wallach et al. 2008]. Ensimmäinen kone-etiikan vaikeuksista onkin eettisten periaatteiden muokkaaminen muotoon, joiden avulla on aina mahdollista tehdä moraalisesti oikeana pidettävä päätös. Jo ensimmäinen ongelma sisältää itsessään useita alaongelmia, sillä harvoin voidaan yksimielisesti olla samaa mieltä parhaasta moraalisesta toimintamallista. Toinen käytännönläheisempi ongelma on eettisten teorioiden muokkaaminen formaaleiksi lausekkeiksi, jotka tekoälyohjelman on mahdollista ymmärtää. Lisäksi on pohdittu, kuka on vastuussa moraalisen tekoälyllisen agentin toiminnan seurauksista, jos sen huomataan toimineen eettisesti väärin [Allen et al. 2006; Hevelke and Nida-Rümelin 2015]. Tässä luvussa käsitellään edellisiä kone-etiikan kysymyksiä sekä niiden merkitystä autonomisien ajoneuvojen kannalta.

4.1. Eettinen rajapinta

Eettisen ulottuvuuden lisäämisen ongelmallisuuteen liittyy kysymys siitä, onko etiikka ylipäätään sovellettavissa tekoälyn ymmärrettäväksi [Anderson and Anderson 2007]. Moor [2006] jakaakin moraaliset agentit eksplisiittisiin ja implisiittisiin, eli tietoisiin ja ei-tietoisiin eettisiin agentteihin niiden ymmärryskyvyn mukaan. Eksplisiittinen eettinen agentti pystyisi ymmärtämään etiikan teorioita, soveltamaan niitä ja täten päätymään parhaaseen mahdolliseen lopputulokseen. Implisiittinen agentti sen sijaan noudattaa siihen ohjelmoituja ohjesääntöjä, eikä sen rakenne mahdollista sääntöjen rikkomista.

Usein moraalisisista autonomisista agenteista puhuttaessa mainitaan Isaac Asimovin vuonna 1942 ilmestyneessä tieteisfiktionovellissa Runaround esitetyt Kolme robotiikan pääsääntöä. Säännöt on suomentanut Matti Kannosto ja ne kuuluvat seuraavasti:

1. Robotti ei saa vahingoittaa ihmistä eikä laiminlyönnin johdosta saattaa tätä vahingoittumaan.

2. Robotin on toteltava ihmisen sille antamia määräyksiä paitsi milloin ne ovat ristiriidassa ensimmäisen pääsäännön kanssa.
3. Robotin on varjeltava omaa olemassaoloaan niin kauan kuin tällainen varjeleminen ei ole ristiriidassa ensimmäisen eikä toisen pääsäännön kanssa.

Implisiittisesti sääntöjä toteuttava ajoneuvo ei pystyisi toimimaan liikenteessä, sillä sen olisi mahdotonta taata itsensä tai liikennettä käyttävien ihmisten turvallisuus. Tieliikenteessä kulkeminen huomattavilla nopeuksilla ei ole ikinä täysin riskitöntä [Goodall 2014], minkä vuoksi ensimmäistä sääntöä noudattava implisiittinen eettinen agentti ei olisi käytännöllinen täysautonomisen ajoneuvon toteutus. Eksplisiittinen agentti puolestaan voisi käyttää Asimovin sääntöjä arvioimalla tilannekohtaisesti kuinka todennäköistä on että sääntöjä jouduttaisiin rikkomaan. Tätä toteutustapaa varten ajoneuvon ja sen valmistajan on tarpeen määritellä millaisia riskejä hyväksytään otettavan, ja miten riskit priorisoidaan onnettomuustilanteessa sen osapuolien välillä [Goodall 2014]. Eksplisiittisen moraalisen agentin luomista pidetään yleisesti kone-etiikan tärkeimpänä päämääränä [Anderson and Anderson 2007], ja siksi tässä tutkielmassa puhutaan ajoneuvoista eksplisiittisinä agentteina.

Vaikka Asimovin lait ovatkin yleisesti kone-etiikassa käytetty esimerkki, ainoastaan niihin nojaaminen eettisiä päätöksiä tehdessä ei ole riittävä eettinen viitekehys. Allen ja muut [2000] muistuttavat, että jo ensimmäinen sääntö voi saattaa järjestelmän lukkiutumistilaan, jos jokainen mahdollinen toimintamalli johtaa oletetusti ihmisen vahingoittumiseen. Tällainen tilanne ei ole tieliikenteessä harvinainen.

Ehkä suurin moraalisen agentin luomiseen liittyvä ongelma on kysymys siitä, miten eettiset periaatteet voitaisiin esittää tekoälyn ymmärtämässä muodossa. Tähän ongelmaan Wallach ja muut [2008] esittävät kokoavia (bottom-up) ja osittavia menetelmiä (top-down).

Kokoava järjestelmä oppii eettisiä toimintatapoja evoluutio- tai oppimisalgoritmien avulla. Se toimii parhaiten, kun tiedetään yksi selvä päämäärä, joka halutaan saavuttaa. Menetelmän heikkous on, että uusiin tilanteisiin, ympäristöihin ja konteksteihin sopeutuminen vie aikaa, jonka vuoksi kokoava menetelmä ei yksinään ole paras vaihtoehto ajoneuvoagentin eettiseen opastukseen.

Osittavassa menetelmässä järjestelmä pyrkii noudattamaan jotain olemassa olevaa eettistä teoriaa, esimerkiksi Asimovin pääsääntöjä. Valittu eettinen teoria jaetaan hierarkkisesti pienempiin osiin, joiden avulla pystytään saavuttamaan haluttu lopputulos. Seuraavaksi esitellään muutamia eettisiä teorioita, joi-

ta pidetään mahdollisina osittavalla menetelmällä agentteihin implementoitavina teorioina.

Ensimmäinen kone-etiikan kirjallisuudessa suosittu eettinen suuntaus on utilitaristinen etiikka. Allen ja muut [2000] määrittelevät utilitarismin näkemykseksi, jonka mukaan lukuisista vaihtoehdoista paras toimintatapa on se, josta saadaan suurin yhteenlaskettu hyöty, parhaimmat seuraukset. Utilitarismi voidaan nähdä tekoälylle sopivana eettisenä teoriana, sillä se vaatii laskentatehoa. Tekoäly pystyy määrittämään jokaisen mahdollisen teon jokaisen mahdollisen seurauksen, ja näin määrittämään parhaan mahdollisen toimintatavan ihmistä tehokkaammin [Anderson and Anderson 2007].

Teoriana utilitarismi vaikuttaa houkuttelevalta, mutta tekoälyn kannalta on kysymyksenä yhä, miten erilaiset seuraukset priorisoidaan ja miten kerätään tarpeeksi informaatiota, jotta voidaan laskea virheetön maksimaalinen hyöty. Jos tavoite on esimerkiksi minimoida ihmisille aiheutuvat vammat, utilitaristinen agentti suosittelisi onnettomuustilanteessa törmäämään pyöräilijään, jolla on kypärä kypärättömän sijasta, sillä kypärätön pyöräilijä saisi todennäköisemmin aivovamman kuin kypärää käyttävä. Tämä päätöspolitiikka olisi kovin epäoikeudenmukainen kypärää käyttäviä pyöräilijöitä kohtaan. [Goodall 2014.]

Toinen moraalisten agenttien yhteydessä usein mainittava eettinen teoria on deontologinen etiikka eli velvollisuusetiikka. Velvollisuusetiikka nimensä mukaan tarkoittaa, että järjestelmään ohjelmoidaan sääntöjä, joiden noudattaminen on sen velvollisuus [Goodall 2014]. Sen mukaan kaikista pätevistä moraalisista velvollisuuksista voidaan muodostaa yksiselitteinen sääntö, moraalinen imperatiivi, jota tulee noudattaa kaikissa tilanteissa [Wallach et al. 2008]. Implisiittisesti toteutettu agentti noudattaisi velvollisuusetiikkaa niiden ominaisuuksien osalta, jotka on määritelty sen velvollisuuksiksi. Toisin kuin implisiittisellä agentilla, velvollisuusetiikkaa noudattavalla agentilla voisi olla myös ymmärrystä eettisistä arvioista, joiden avulla se muodostaisi yhdessä velvollisuuksiensa kanssa päätelmiä oikeista toimintatavoista.

Velvollisuusetiikan ongelma on sama kuin implisiittisen agentinkin, eli ainoastaan sääntöjä rikkovien toimintamahdollisuuksien aiheuttama lukkiutuminen [Allen et al. 2000]. Toinen suuri ongelma on, että halutuista säännöistä on vaikea muodostaa tekoälyn ymmärtämiä moraalisia imperatiiveja, sillä useimpien sääntöjen noudattamisessa tarvitaan myös maalaisjärkeä. Esimerkiksi Asimovin sääntöjä noudattava ajoneuvoagentti voisi jättää hätäjarrutuksen tekemättä, koska sen äkillinen jarrutus saattaisi aiheuttaa matkustajille niskavamman. [Goodall 2014.]

Kolmantena eettisenä teoriana esitellään hyve-etiikka. Hyve-etiikan mukaan ennalta määrättyjen sääntöjen noudattamisen sijaan tulisi oppia moraali-

sesti tärkeitä hyveitä, kuten rehellisyys tai myötätunto, joita harjoittamalla pystytään tekemään moraalisesti oikea päätös [Allen et al. 2000]. Hyve-etiikan vahvuus on, ettei se rajoita toimintaa yksittäisiin tilanteisiin, vaan mahdollistaa moraalisen arvioinnin myös uusissa tilanteissa ja ympäristöissä [Anderson and Anderson 2007].

Allen ja muut [2000] esittävät hyve-etiikan sisältävän samoja ongelmia kuin velvollisuusetiikan, sillä kuten velvollisuudet myös hyveet voivat olla ristiriidassa keskenään. Heidän mielestään hyve-etiikkaa ei kuitenkaan pitäisi rinnastaa velvollisuusetiikkaan, sillä hyve-etiikka mahdollistaa velvollisuuksista erillisen moraalisen arviointiperusteen. Hyve-etiikan implementointi tekoälyagenttiin on kuitenkin erittäin haasteellista, sillä hyveet ilmenevät ihmiskäytöksessä epäsuorasti monimutkaisten psykologisten prosessien seurauksena [Wallach et al. 2008].

Hyve-etiikan kannattajien kesken on erimielisyyksiä siitä, mitkä ominaisuudet luetaan hyveiksi [Wallach et al. 2008]. Samoin on vaikeaa keksiä esimerkkejä hyveistä, joiden perusteella autonominen ajoneuvo pystyisi tekemään moraalisia päätöksiä. Yksi ajoneuvon hyveistä voisi olla esimerkiksi uskollisuus, jolloin se priorisoisi matkustajiensa hyvinvoinnin muiden liikenteenkäyttäjien yläpuolelle. Toisaalta myös oikeudenmukaisuus voidaan nähdä hyveenä, jolloin tarvitaan syvempää analyysiä siitä, toteuttaako uskollisuus oikeudenmukaisuuden hyvettä vai ovatko ne ristiriidassa keskenään.

Goodall [2014] uskoo, että paras moraalinen agentti saavutetaan yhdistämällä kaksi tai useampi eettinen teoria. Myös Allen ja muut [2000] ovat sitä mieltä, että esimerkiksi velvollisuusetiikkaa ja utilitarismia noudattava agentti voisi toimia velvollisuuksiensa mukaisesti, paitsi jos voidaan utilitaristisen etiikan mukaan osoittaa, että velvollisuuden rikkomisesta saadut seuraukset ovat selvästi sen noudattamista kannattavampia. Lisäksi Anderson ja Anderson [2007] huomioivat, että velvollisuuksien noudattamisen lisäksi agentin on tärkeää ottaa huomioon tekojen seuraukset sekä niiden oikeudenmukaisuus. Useita teorioita ymmärtävän agentin uskotaan pääsevän todennäköisemmin moraalisesti perusteltuun lopputulokseen kuin vain yhtä teoriaa noudattavan, sillä ne tuntuvat täydentävän toisiaan.

Utilitaristiseen etiikan heikkous on, ettei se ota huomioon tilannekontekstia tai yksilön oikeuksia [Goodall 2014]. Jos mietitään onnettomuuskontekstia, usein voi olla tarpeen arvioida myös sitä kenen virhearviosta onnettomuus syntyy. Oikeudenmukaisuutta hyveenä pitävä hyve-eettinen ajoneuvoagentti voisi tällöin laskelmoida, että minimaalisen vahingon sijaan aiheutetaan vahinkoa onnettomuuden syylliselle osapuolelle. Ajoneuvolla voisi kuitenkin olla vielä velvollisuutenaan välttää hengenvaaralliset onnettomuudet sen ollessa mahdol-

lista. Jos siis uskotaan, että syyntakeinen on erittäin suuressa hengenvaarassa, mutta toisaalta onnettomuuteen syytön osapuoli pystyy selviämään tilanteesta pienin vahingoin, voitaisiin päätyä onnettomuuden lopputuloksen kannalta optimaalisimpaan tilanteeseen kolmea tässä luvussa esiteltyä eettistä teoriaa käyttäen.

Wallachin ja muiden [2008] mukaan ihmisen moraali perustuu sekä omiin kokemuksiin että teoreettiseen järkeilyyn. Heidän mielestään paras lopputulos myös kone-etiikassa saavutetaan yhdistämällä osittavilla menetelmillä implementoidut eettiset teoriat ja kokoavien menetelmien kyky oppia soveltamaan sääntöjä joustavasti. Menetelmästä riippumatta säilyy kysymys siitä, miten monimutkaiset eettiset teoriat voidaan kääntää tekoälyn ymmärtämään muotoon. Anderson ja Anderson [2007] uskovat, että kone-etiikan tutkimus edistää myös eettisen teorian tutkimusta, sillä se edellyttää eettisten sääntöjen muokkaamista yksiselitteisiksi ohjenuoriksi. Myös Moor [2006] kokee, että kone-etiikka tulee edistämään käsitystämme etiikasta, mutta uskoo, ettei tekoäly pysty eettiseen päätöksentekoon vielä lähitulevaisuudessa. Hänen mielestään on kuitenkin tärkeää jatkaa kone-etiikan kysymysten pohtimista jatkoa ajatellen, sillä hän uskoo teknologian saavuttaman autonomisuuden ainoastaan lisääntyvän. Ajoneuvoagenttien autonomisuuden kasvaessa ja ajovastuun siirtyessä yhä enemmän ihmiseltä pois on syytä pohtia vastuunäkökulmaa myös ajoneuvon täysautonomisuuden kannalta.

5. Vastuunäkökulma

Kolmannessa luvussa käsiteltiin auktoriteetin jakoa ihmisen ja ajoneuvoagentin välillä. Tässä luvussa pohditaan, voisiko tämänhetkinen vastuuasetelma vaihtua, mikäli autonomiseen ajoneuvoon onnistutaan implementoimaan eettisiin päätöksiin kykenevä rajapinta.

Kuten on jo aiemmin todettu, tällä hetkellä ajoneuvoagentti ei saa liikkua tieliikenteessä ilman ihmisvalvojaa [Gurney 2015]. Vastuu ajoneuvosta on aina valvojalla, mutta esimerkiksi Goodallin [2014] mukaan on kohtuutonta odottaa, että autonomista ajoneuvoa valvova ihminen pystyisi vaaratilanteessa tekemään tilanteen vaatimat eettiset päätökset lyhyellä varoitusajalla. Tästä näkökulmasta vastuu itse eettisestä toimintatavasta olisi jo ajoneuvoagentilla, vaikka päätöksen seurauksista vastuuseen joutuukin ihminen. Onkin aiheellista pohtia, voiko ihminen olla vastuussa täysautonomisen ajoneuvoagentin päätöksistä, vaikka hänellä ei itsellään olisi osuutta niihin. Samoin on syytä miettiä, ketä pidetään vastuussa, jos autonominen agentti suorittaa haitallisia tai laittomia toimintoja [Allen et al. 2006]. Onko vastuu itse ajoneuvolla vai jollain muulla?

Allenin ja muiden [2000] mukaan järjestelmää, joka ei ymmärrä tekojensa seurauksia, ei voida pitää syyllisenä tekojensa seurauksiin. Samoin Anderson ja Anderson [2007] pitävät moraalisen vastuun edellytyksenä tekijän vapaata tahtoa sekä teon tarkoituksenmukaisuutta. Kysymys siitä, voiko tekoälyllisellä agentilla olla vapaata tahtoa tai todellista ymmärrystä tekojensa seurauksista, on monimutkainen ja myös tämän tutkielman kontekstissa epäoleellinen: ajoneuvoa voidaan pitää eettisesti vastuullisena tekemistään vahingoista, mutta se on täysin kykenemätön maksamaan vahingonkorvauksia. Tässä mielessä taloudellisesti vastuussa olevan tahon on aina oltava ihminen.

Suomessa onnettomuuden aiheuttanut osapuoli maksaa itselleen sekä syyttömälle asianosaiselle koituneet henkilö- ja omaisuusvahingot [Liikennevakuutuslaki 279/1959], mutta syyllisen ollessa autonominen ajoneuvo lain tulkitseminen vaikeutuu. Yksi tulkinta voisi olla, että korvausvelvollisuus lankeaa onnettomuuden aiheuttaneen autonomisen ajoneuvon matkustajille. Toisen näkemyksen mukaan syyllinen ajoneuvon aiheuttamiin onnettomuuksiin on sen valmistaja.

5.1. Käyttäjän vastuu

Gurney [2013] ajattelee, että auton matkustajan vastuullisuus onnettomuustilanteessa on riippuvainen matkustajan kyvystä estää onnettomuus. Jos autonomiset ajoneuvot halutaan tuoda liikuntarajoitteisten ihmisten käytettäviksi, olisi kohtuutonta odottaa, että esimerkiksi sokea matkustaja pystyisi estämään ajoneuvon onnettomuuden. Samoin täysautonomisuuden etuna voidaan pitää kuljettajan vapauttamista ajotehtävistä, jolloin olisi tarkoituksenvastaista edellyttää ajoneuvon matkustajalta jatkuvaa liikenteen valvontaa. Jos kuitenkin todetaan, että ajamiseen kykenevä matkustaja on omalla syötteellään tai järjestelmän varoitukset laiminlyömällä myötävaikuttanut vahinkoon, voidaan häntä pitää vastuussa onnettomuuden seurauksista. Lisäksi ajoneuvovalmistaja voi esittää tuotteelleen riskejä vähentäviä rajoituksia, kuten ettei auto saa ajaa autonomisesti lumisateella. Jos näitä sääntöjä rikotaan, matkustaja on jälleen vastuussa mahdollisista onnettomuuksista.

Hevelke ja Nida-Rümelin [2015] esittävät kaksi tapaa pitää vastuu ihmisellä. Ensimmäinen tapa on pysyä nykyisessä mallissa, jossa ihmisellä on – kuten aiemmin todettua – aina vastuu ajoneuvostaan. Tämä perustelu kuitenkin tuhoaisi yhden täysautonomisuuden tärkeimmistä päämääristä, kuljettajan vapauttamisen ajotehtävistä, mikä mahdollistaisi esimerkiksi liikuntarajoitteisten yksityisautoilun. Näkemys on kuitenkin perusteltavissa, jos todetaan, että ihmisen valvonnan alla operoiva auto on huomattavasti turvallisempi kuin täysautonominen ajoneuvo.

Toinen tapa on käyttää niin sanottua ankaraa vastuuta, eli pitää kuljettaja vastuussa autonomisen ajoneuvon virheistä, vaikka hänellä ei olisi niihin mitään osuutta. Tieliikenteessä ajamiseen liittyy aina onnettomuuden mahdollisuus, jonka ajoneuvon käyttäjät tiedostivat. Tällä perustella kuljettaja tai matkustaja ottivat riskin, ja täten myös vastuun, jo suostuessaan autonomisen ajoneuvon kyydittäviksi.

Voidaan kuitenkin pohtia, että jos valmistajilla ei ole mitään vastuuta autonomisen järjestelmän toiminnasta, miten voidaan varmistaa niiden maksimaalinen turvallisuus? Gurneyn [2013] mielestä valmistajien pitäminen vastuussa varmistaisi, että he tekevät kaikkensa tuotteidensa turvallisuuden eteen. Samoin Hevelke ja Nida-Rümelin [2015] uskovat valmistajien vapauttamisen vastuusta johtavan puutteisiin ajoneuvojen turvallisuudessa.

5.2. Valmistajan vastuu

Voi olla, että ajovastuun siirtyessä pois ihmiseltä myös vastuu onnettomuuksista alkaa väistyä kuljettajalta ajoneuvon valmistajille [Marchant and Lindor 2012]. Esimerkiksi Gurneyn [2013] mielestä ajoneuvovalmistajan tulisi olla vastuussa onnettomuuksista, jotka tapahtuvat autonomisessa ajossa. On kuitenkin olemassa aiheellinen epäilys siitä, että vastuu onnettomuuksista voi vähentää ajoneuvovalmistajien halukkuutta kehittää autonomisia järjestelmiä [Marchant and Lindor 2012].

Huolimatta siitä, kuinka turvallisia autonomisista ajoneuvoista pystytään tekemään, ajoneuvoagentteja kohtaan tullaan olemaan paljon ihmiskuskia kriittisempiä [Fagnant and Kockelman 2015]. Yksittäisten onnettomuuksien uutisointi mediassa voi luoda vääristyneen kuvan autonomisien ajoneuvojen turvallisuudesta. Valmistajan kannalta jokaisen onnettomuuden estäminen on mahdotonta, sillä suurin osa onnettomuuksista tulee tapahtumaan tilanteissa, joita on vaikea ennakoida tuotetta suunnitellessa [Marchant and Lindor 2012]. Lisäksi ajoneuvo voi toimia tilanteessa optimaalisesti ja silti aiheuttaa vahinkoja, jos kaikki mahdolliset vaihtoehdot ovat huonoja. On ehdotettu, että autonomisien ajoneuvojen tulisi säilyttää tietty määrä lokitietoja, joiden perusteella voidaan tarkistaa sen tekemien päätöksien perustelu sekä mahdollinen kuljettajan osuus onnettomuuteen [Fagnant and Kockelman 2015]. Datapankin avulla valmistajat voisivat osoittaa ajoneuvon toimineen oikeutetusti tai kuljettajan myötävaikuttaneen onnettomuuden syntymiseen.

Tällä hetkellä näyttää siltä, että valmistajien vastuu onnettomuuksista voi toimia merkittävänä esteenä autonomisten ajoneuvojen kehittämiselle, vaikka niiden uskotaan olevan vaikutuksiltaan hyvinvointia parantavia [Marchant and Lindor 2012]. Hevelke ja Nida-Rümelin [2015] esittävätkin, että vastuunalaisuus liikenteessä tulisi olla vain osittain ajoneuvovalmistajilla, jotta kehitystyö kan-

nattaa, mutta ei kuitenkaan uhkaa yrityksiä liikaa. Lainsäätäjien on jo korkea aika alkaa pohtia täysautonomisten ajoneuvojen aiheuttamiin onnettomuuksiin liittyviä tilanteita, jotta voidaan varmistaa, että vastuullinen osapuoli on selkeäsi tiedossa tulevaisuudessa, kun täysautonomiset ajoneuvot saapuvat liikenteeseen [Gurney 2013]. Lainsäädännön on tärkeä toimia nopeasti, sillä ominaisuuksiltaan puutteellisillakin autonomisilla ajoneuvoilla on potentiaalia tulla nykyisiä turvallisemmiksi ja pystyä parantamaan liikenneturvallisuutta [Marchant and Lindor 2012], eli säästämään ihmishenkiä.

6. Yhteenveto

Ajoneuvoihin pyritään toteuttamaan yhä enemmän autonomisia ominaisuuksia, joilla pyritään tehostamaan tieliikennettä, vähentämään kasvihuonepäästöjä sekä ennen kaikkea lisäämään turvallisuutta. Edellisistä jaloista päämääristä jälkimmäisin on kaikkein kriittisin, ja siinä epäonnistuminen voi estää autonomisien ajoneuvojen lopullisen läpimurron tieliikennekäytössä. Tämänhetkisen ajoneuvokannan sekä lainsäädännön vaatima ihmiskuljettaja voi tulevaisuudessa joutua väistymään, mikäli nämä autonomisoinnin päämäärät pystytään saavuttamaan. Suurempi turvallisuus vaatii tekoälyltä kuitenkin myös päätöksentekokykyä vaikeissa tilanteissa, jonka saavuttaminen edellyttää vielä ponnistelua kone-etiikan alalla. Jos eettisiä teorioita kuten utilitarismi, velvollisuusetiikka ja hyve-etiikka pystytään kääntämään tekoälyn ymmärtämään muotoon luoden moraalisesti toimivia agentteja, ollaan jo lähellä onnistumista. Jäljelle jää kuitenkin vielä lainsäädännöllinen kysymys siitä, kuka ottaa vastuun moraalisen ajoneuvoagentin toiminnasta. Lainsäätäjien sekä autovalmistajien välinen dialogi jo kehityksen aikaisessa vaiheessa todennäköisesti edistäisi autonomisien järjestelmien implementaatiota.

Useat autonomisia ajoneuvojärjestelmiä koskevat kysymykset kaipaavat yhä vastauksia ennen kuin niiden laaja käyttöönotto on ajankohtaista. Tutkielman perusteella lienee turvallista väittää, että tulevaisuuden itseohjautuvien autojen suunnittelussa valmistajien kannattaa kiinnittää huomiota myös ajoneuvon käyttöön liittyviin eettisiin ongelmiin.

Viiteluettelo

Colin Allen, Gary Varner and Jason Zinser. 2000. Prolegomena to any future artificial moral agent. *Journal of Experimental & Theoretical Artificial Intelligence* 12, 3, 251-261.

Colin Allen, Wendell Wallach and Iva Smit. 2006. Why machine ethics? *IEEE Intelligent Systems,* 21, 4, 12-17.

Michael Anderson and Susan Leigh Anderson. 2007. Machine ethics: creating an ethical intelligent agent. *AI Magazine* 28, 4, 15-26.

Victoria A. Banks and Neville A. Stanton. 2015. Keep the driver in control: automating automobiles of the future. *Applied Ergonomics* 53, 389-395.

Olivier Boissier, Grégory Bonnet, Jean-Gabriel Ganascia, Catherine Tessier, Thibault de Swarte and Robert Voyer. 2015. A roadmap towards ethical autonomous agents. *ANR ETHICAA –ANR-13-CORD-0006*, 3. Saatavilla osoitteessa: https://ethicaa.greyc.fr/media/files/ethicaa.delivrable.3.pdf, viitattu 18.2.2016.

Daniel J. Fagnant and Kara Kockelman. 2015. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice* 77, 7, 167-181.

Stan Franklin and Art Graesser. 1996. Is it an agent, or just a program?: A taxonomy for autonomous agents. In: *Intelligent Agents III. Agent Theories, Architectures, and Languages*, 21-35.

Noah J. Goodall. 2014. Machine ethics and automated vehicles. *Road Vehicle Automation*, 93-102

Google. 2015. *Google Self-Driving Car Testing Report on Disengagements of Autonomous Mode December 2015.*

Google. 2016. *Google Self-Driving Car Project Monthly Report February 2016.*

Erico Guizzo. 2011. How google's self-driving car works. *IEEE Spectrum Online* 11.

Jeffrey K. Gurney. 2013. Sue my car not me: products liability and accidents involving autonomous vehicles. *University of Illinois Journal of Law, Technology & Policy*, 247.

Alexander Hevelke and Julian Nida-Rümelin. 2015. Responsibility for crashes of autonomous vehicles: an ethical analysis. *Science and Engineering Ethics* 21, 3, 619-630.

Nicholas R. Jennings. 2000. On agent-based software engineering. *Artificial Intelligence* 117, 2, 277-296.

Liikennevakuutuslaki 279/1959.

Gary E. Marchant and Rachel A. Lindor. 2012. The coming collision between autonomous vehicles and the liability system. *Santa Clara Law Review* 52, 4, Article 6.

Natasha Merat and A. Hamish Jamson. 2009. How do drivers behave in a highly automated car. In: *Proceedings of the 5th International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*, 514-521.

James H. Moor. 2006. The nature, importance, and difficulty of machine ethics. *IEEE Intelligent Systems* 21, 4, 18-21.

William Payre, Julien Cestac, Patricia Delhomme. 2014. Intention to use a fully automated car: attitudes and a priori acceptability. *Transportation Research Part F: Traffic Psychology and Behaviour* 27, 11, 252–263.

Dale Richards and Alex Stedmon. 2015. To delegate or not to delegate: a review of control frameworks for autonomous cars. *Applied Ergonomics* 53, 383-388.

Sivistyssanakirja, Suomisanakirja. 2015. Saatavilla osoitteessa: http://www.suomisanakirja.fi/autonomia, viitattu 9.2.2016.

Tilastokeskus. 2014. Tieliikenneonnettomuustilasto. Saatavilla osoitteessa: http://www.stat.fi/til/ton/2014/ton_2014_2015-12-16_tie_001_fi.html, viitattu 23.2.2016.

Wendell Wallach, Colin Allen and Iva Smit. 2008. Machine morality: bottom-up and top-down approaches for modelling human moral faculties. *AI & Society* 22, 4, 565-582.